L Number	Hits	Search Text	DB	Time stamp
1	396	713/168.ccls.	USPAT	2004/03/02 09:59
2	102	713/168.cor.	USPAT	2004/03/02 15:20
3	0	jakobsson-bjorn	USPAT	2004/03/02 09:59
4	0	jakobsson-bjorn.in.	USPAT	2004/03/02 09:59
5	1	jakobsson-markus.in.	USPAT	2004/03/02 10:00
6	9	(jakobsson near bjorn near markus).in.	USPAT	2004/03/02 10:02
7	5	juels-ari.in.	USPAT	2004/03/02 10:05
8	5	(juels near ari).in.	USPAT	2004/03/02 10:09
9	0	bread near pudding near protocol	USPAT	2004/03/02 10:10
10	0	bread near pudding near protocol	USPAT	2004/03/02 10:10
11	16	bread near pudding	USPAT	2004/03/02 10:10
12	42	proof near work	USPAT	2004/03/02 10:25
13	6	(proof near work) and encrypt$	USPAT	2004/03/02 10:11
14	0	(proof near work) and POW	USPAT	2004/03/02 10:31
15	936	distribut$ near (task or job)	USPAT	2004/03/02 10:27
16	0	(proof near work) and (distribut$ near (task or job))	USPAT	2004/03/02 10:26
17	718	distribut$ near task	USPAT	2004/03/02 10:27
18	15	(distribut$ near task) and (digital near signature)	USPAT	2004/03/02 10:28
19	0	((distribut$ near task) and (digital near signature)) and (sub near task)	USPAT	2004/03/02 10:28
21	0	((distribut$ near task) and (sub near task)) and (digital near signature)	USPAT	2004/03/02 10:28
20	30	(distribut$ near task) and (sub near task)	USPAT	2004/03/02 10:28
22	0	(proof near work) and ((distribut$ near task) and (sub near task))	USPAT	2004/03/02 10:32
23	80	task same (digital near signature)	USPAT	2004/03/02 10:32
24	28	task WITh (digital near signature)	USPAT	2004/03/02 10:33
25	0	(proof near work) WITh (digital near signature)	USPAT	2004/03/02 10:33
26	0	(proof near work) same (digital near signature)	USPAT	2004/03/02 10:33
27	1	(proof near work) and (digital near signature)	USPAT	2004/03/02 10:46
28	321	verif$ near task	USPAT	2004/03/02 10:46
29	10	(verif$ near task) and (digital near signature)	USPAT	2004/03/02 10:48
30	1	((distribut$ near task) and (sub near task)) and proof	USPAT	2004/03/02 10:49
31	7	((distribut$ near task) and (sub near task)) and verif$	USPAT	2004/03/02 10:52
32	6	task near activity near log	USPAT	2004/03/02 11:42
33	1	(task near activity near log) and restrict$	USPAT	2004/03/02 10:57
34	0	(task near activity near log) and prohibit$	USPAT	2004/03/02 10:57

35	106	minting	USPAT	2004/03/02 11:43
36	10	minting and hash$	USPAT	2004/03/02 13:21
37	6	(minting and hash$) and (task or work)	USPAT	2004/03/02 11:44
38	0	(minting and hash$) and (sub near task)	USPAT	2004/03/02 11:44
39	2	(minting and hash$) and image	USPAT	2004/03/02 11:46
40	0	minting WITH partition$	USPAT	2004/03/02 11:48
41	0	minting same partition$	USPAT	2004/03/02 11:47
42	7	minting and partition$	USPAT	2004/03/02 11:47
43	0	minting WITH sub	USPAT	2004/03/02 11:47
45	0	(minting WITH (partition$ or divid$ or separat$ or portion$ or segment$ or split$ or division$)) and hash$	USPAT	2004/03/02 11:50
44	7	minting WITH (partition$ or divid$ or separat$ or portion$ or segment$ or split$ or division$)	USPAT	2004/03/02 12:52
47	2	(partition$ WITH mint$) and hash$	USPAT	2004/03/02 12:41
46	29	partition$ WITH mint$	USPAT	2004/03/02 12:51
48	52	distribut$ WITH mint$	USPAT	2004/03/02 12:51
49	278	mint$ WITH (partition$ or divid$ or separat$ or portion$ or segment$ or split$ or division$)	USPAT	2004/03/02 12:53
50	0	mint$ WITH (sub near task)	USPAT	2004/03/02 12:53
51	0	mint$ same (sub near task)	USPAT	2004/03/02 12:53
52	5	mint$ and (sub near task)	USPAT	2004/03/02 12:53
53	10	hash$ WITH range WITH image	USPAT	2004/03/02 13:24
54	25098	least near significant near bit	USPAT	2004/03/02 13:25
55	2	(hash$ WITH range WITH image) and (least near significant near bit)	USPAT	2004/03/02 13:58
56	2	((hash$ WITH range WITH image) and (least near significant near bit)) and advantag$	USPAT	2004/03/02 13:46
57	1	6549210.pn.	USPAT	2004/03/02 13:47
58	11	search$ WITH different WITH space WITH solution	USPAT	2004/03/02 14:01
59	0	(search$ WITH different WITH space WITH solution) and mint$	USPAT	2004/03/02 14:00
60	0	(search$ WITH different WITH space WITH solution) and hash$	USPAT	2004/03/02 14:00
61	1	(search$ WITH different WITH space WITH solution) same advantag$	USPAT	2004/03/02 14:01
62	9	(search$ WITH different WITH space WITH solution) and advantag$	USPAT	2004/03/02 14:13
63	1	valid with solution with coin	USPAT	2004/03/02 14:18
64	1085	valid with solution	USPAT	2004/03/02 14:14
65	1	(valid with solution) same coin	USPAT	2004/03/02 14:14
66	2	secret near value with coin	USPAT	2004/03/02 14:45
67	1	5768385.pn.	USPAT	2004/03/02 14:45

68	1	5768385.pn. and advantag$	USPAT	2004/03/02 15:02
69	1	5768385.pn. and secret	USPAT	2004/03/02 15:07
70	0	5768385.pn. and (secret with hash$)	USPAT	2004/03/02 15:03
71	0	5768385.pn. and (secret same hash$)	USPAT	2004/03/02 15:03
73	0	(hash$ WITH concatenat$ WITH secret) same advantag$	USPAT	2004/03/02 15:08
74	52	(hash$ WITH concatenat$ WITH secret) and advantag$	USPAT	2004/03/02 15:09
75	22	((hash$ WITH concatenat$ WITH secret) and advantag$) and coin	USPAT	2004/03/02 15:08
76	22	(hash$ WITH concatenat$ WITH secret) and coin	USPAT	2004/03/02 15:09
72	58	hash$ WITH concatenat$ WITH secret	USPAT	2004/03/02 15:09
77	31	713/180.cor.	USPAT	2004/03/02 15:20
78	570	713/200.cor.	USPAT	2004/03/02 15:21
79	6	713/200.cor. and coin	USPAT	2004/03/02 15:21
80	19	713/200.cor. and monetary	USPAT	2004/03/02 15:21
81	94	705/67-69,76.cor.	USPAT	2004/03/02 15:22

US005768385A

United States Patent [19]

Simon

[11] Patent Number: 5,768,385

[45] Date of Patent: Jun. 16, 1998

[54] **UNTRACEABLE ELECTRONIC CASH**

[75] Inventor: **Daniel R. Simon**, Redmond, Wash.

[73] Assignee: **Microsoft Corporation**, Redmond, Wash.

[21] Appl. No.: **521,124**

[22] Filed: **Aug. 29, 1995**

[51] **Int. Cl.**[6] **H04L 9/00**; H04L 9/30

[52] **U.S. Cl.** **380/24**; 380/23; 380/25; 380/30; 380/49

[58] **Field of Search** 380/23, 24, 25, 380/29, 30, 49, 59, 4, 9, 50

[56] **References Cited**

U.S. PATENT DOCUMENTS

4,914,698	4/1990	Chaum	380/30
4,947,430	8/1990	Chaum	380/25
4,949,380	8/1990	Chaum	380/30
4,987,593	1/1991	Chaum	380/30 X
4,991,210	2/1991	Chaum	380/30
4,996,711	2/1991	Chaum	390/30
5,131,039	7/1992	Chaum	380/23
5,276,736	1/1994	Chaum	380/24
5,373,558	12/1994	Chaum	380/23

OTHER PUBLICATIONS

A. Pfitzmann, "How to Implement ISDNs Without User Observability—Some Remarks," TR 14/85, Fakultät für Informatik Universität Karlsruhe, 1985.

Okamoto, et al., "Universal Electronic Cash," Proc. CRYPTO 191, Springer-Verlag (1992), pp. 324–337.

Rompel, "One-Way Functions Are Necessary and Sufficient for Secure Signatures," Proc. 31st IEEE Symp. on Foundations of Computer Science (1990), pp. 387–394.

Brands, "Untraceable Off-line Cash in Wallet with Observers" Proc. CRYPTO '93, Springer-Velag (1994) pp. 302–318.

Yacobi, "Efficient Electronic Money," Proc. ASIACRYPT 194, Springer-Verlag (1994).

Rackoff, et al. "Cryptographic Defense Against Traffic Analysis," Proc. 25th ACM Symp. on the Theory of Computation (1993).

Chaum, "Online Cash Checks," Proc. EUROCRYPT '89, Springer-Verlag (1989), pp. 288–293.

Chaum, "The Dining Cryptographers Problem: Unconditional Sender and Recipient Untraceability," Journal of Cryptology, vol. 1, No. 1 (1988), pp. 65–75.

Chaum, "Privacy Protected Payments—Unconditional Payer and/or Payee Untraceability," Smart Card 2000: The Future of IC Cards—Proc. IFIP WG 11.6 Int'l Conf. North-Holland (1989) pp. 69–93.

Pfitzmann, et al. "ISDN-MIXes—Untraceable Communication with Very Small Brandwidth Overhead," Proc. Kommunikation in verteilten Systemen (1991), pp. 451–463.

Even, et al. "Electronic Wallet," Proc. CRYPTO '83, Plenum Press (1984), pp. 383–386.

Chaum, et al. "Untraceable Electronic Cash," Proc. CRYPTO '88, Springerverlag (1990), pp. 319–327.

Franklin, et al., "Secure and Efficient Off-Line Digital Money," Proc. 20th Int'l Colloquim on Automata Languages and Programming, Springer-Verlag (1993), pp. 265–276.

Chaum, "Achieving Electronic Privacy," Scientific American, vol. 267, No. 2 (1992), pp. 96–101.

Chaum, "Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms," CACM, vol. 24, No. 2 (1981) pp. 84–88.

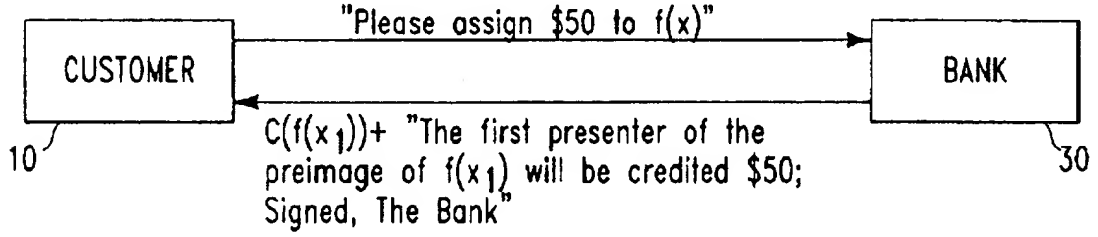
Primary Examiner—Bernarr E. Gregory

Attorney, Agent, or Firm—Michaelson & Wallace; Peter L. Michaelson

[57] **ABSTRACT**

An electronic cash protocol including the steps of using a one-way function $f_1(x)$ to generate an image $f_1(x_1)$ from a preimage x_1; sending the image $f_1(x_1)$ in an unblinded form to a second party; and receiving from the second party a note including a digital signature, wherein the note represents a commitment by the second party to credit a predetermined amount of money to a first presenter of the preimage x_1 to the second party.

30 Claims, 2 Drawing Sheets



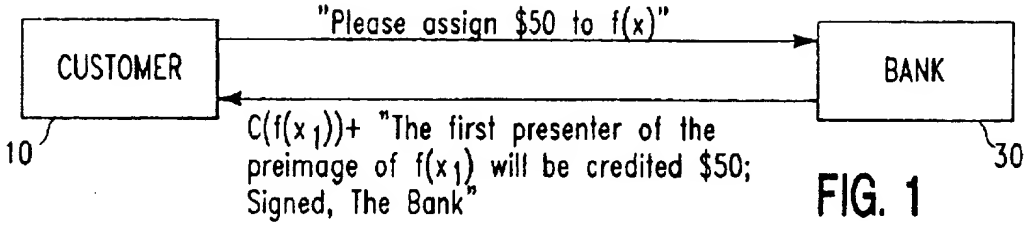


FIG. 1

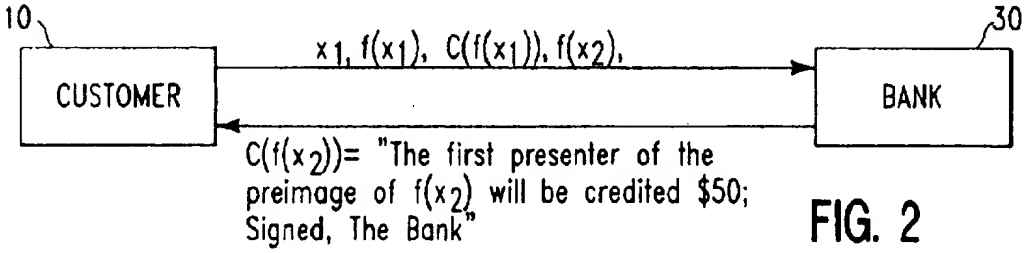


FIG. 2

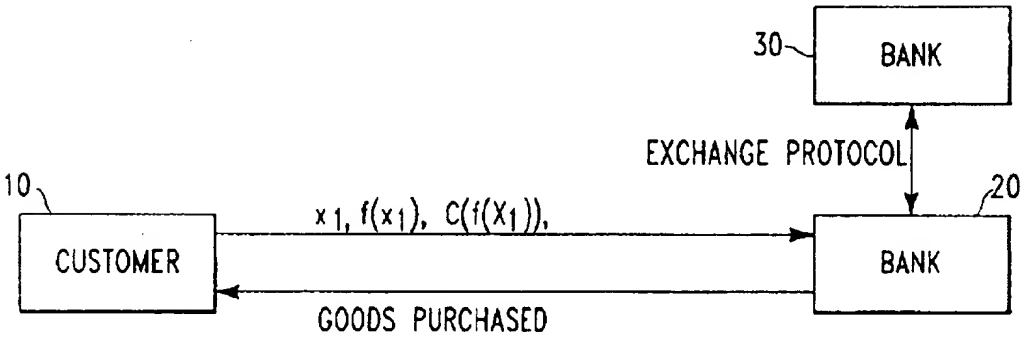


FIG. 3

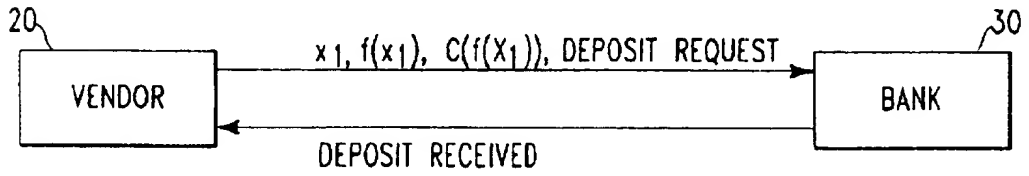


FIG. 4

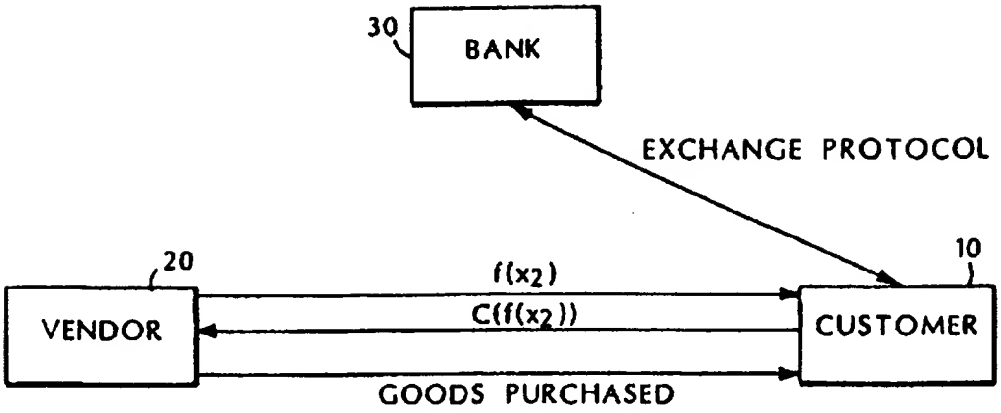


FIG. 5

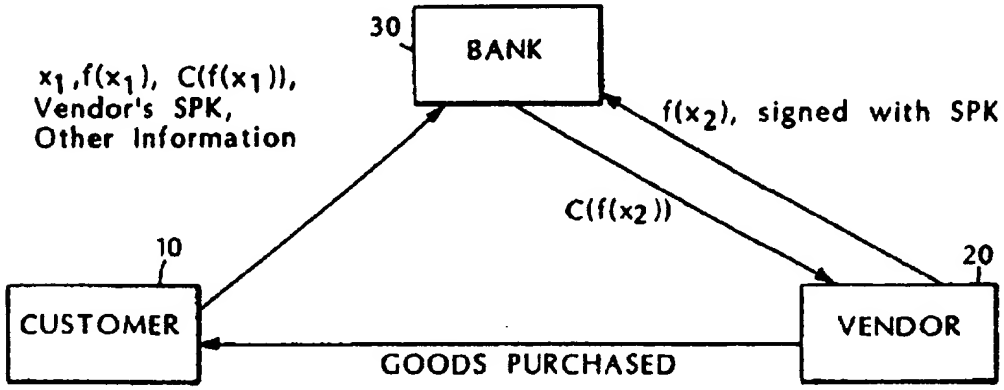


FIG. 6

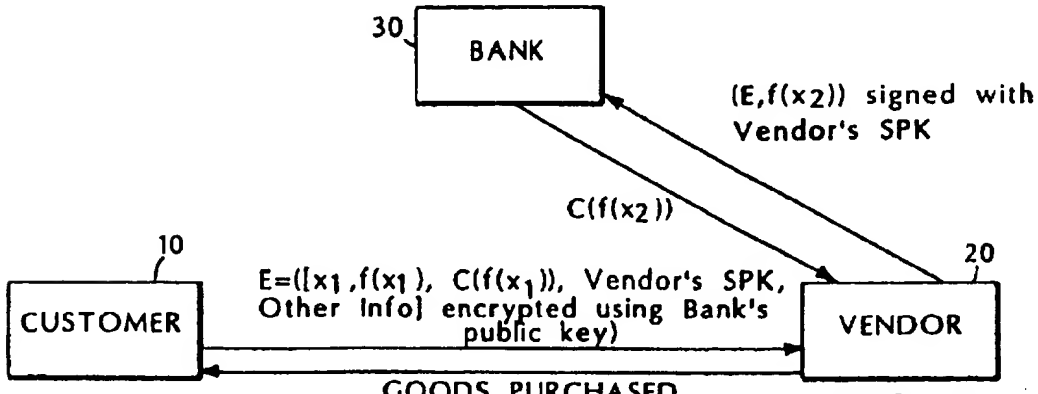


FIG. 7

UNTRACEABLE ELECTRONIC CASH

BACKGROUND OF THE INVENTION

The invention generally relates to electronic cash systems.

The ultimate intuitive goal of an electronic cash system is to combine the best features of physical cash (privacy, anonymity, unforgeability) with the best features of electronic commerce (speed, ease and potential security of transport and storage). The fundamental difficulty with attempting to implement anonymous electronic cash, however, is simple to state: if the possessor of an electronic "coin" is not identified in two successive transactions, then how is he or she to be prevented from acting as if the first transaction never occurred, and spending the same coin again. The first proposed solution to this problem was presented by Chaum, Fiat and Naor (see D. Chaum, A Fiat and M. Naor, Untraceable Electronic Cash, Proc. CRYPTO '88, Springer-Verlag (1990), pp. 319–327.), and was based on the premise that it would be sufficient for such "double spending" to be detected, and the spender identified, upon presentation of the same "electronic coin" twice to the central bank. This premise has also been used in a number of other proposed solution, all with the advantage that the bank need not be involved in each transaction. Practically speaking, however, this premise has enormous drawbacks. Fraudulent transactions are detected only long after they have taken place, and if the perpetrator can be confident of not being brought to justice (either by being inaccessible or by managing to use someone else's identity and cash), then he or she can double-spend at will.

However, if such fraudulent use of electronic cash is to be prevented, then some authority must somehow be involved in each transaction as it occurs, so as to be able to recognize and alert targets of double-spending. How, then, is anonymity to be preserved. One approach is to rely on tamper-resistant hardware to force spenders to behave "honestly" (ie., not to double-spend) (see, for example, S. Even, O. Goldreich and Y. Yacobi, Electronic Wallet, Proc. CRYPTO '83, Plenum Press (1984), pp. 383–386.). Schemes based on this premise are, however, extremely "brittle". If anyone ever succeeds in tampering with the hardware, then not only is that person capable of double-spending, but anyone, anywhere who obtains (e.g. purchases, perhaps) the information hidden in the hardware can spend arbitrarily high amounts at will. Current tamper resistance technology is far from being dependable enough to be trusted to thwart such an enormous risk.

Another approach is cryptographic. For example, under certain very strong cryptographic assumptions, it is possible to construct protocols that create "blinded" cash—information which can be recognized later as valid cash, but cannot be connected with any particular run of the protocol. (See, for example, D. Chaum, Privacy Protected Payments—Unconditional Payer and/or Payee Untraceability, SMART CARD 2000: The Future of IC Cards—Proc. IFIP WG 11.6 Int'l Conf., North-Holland (1989), pp. 69–93; and D. Chaum, Online Cash Checks, Proc. EUROCRYPT '89, Springer-Verlag (1989), pp. 288–293.)

SUMMARY OF THE INVENTION

We present a simple, practical online electronic cash system based on the assumption of a network in which anonymous, untraceable communication is possible. In general, the invention uses two simple primitives, namely a one-way function and a signature scheme. These are both well known in the art; and descriptions can be found in publicly available literature on cryptography, e.g. Applied Cryptography, Bruce Schneier, John Wiley & Sons, Inc. (1994). The anonymity of spenders as well as guaranteeing their electronic coins'validity, but also the coins used are unforgeable and cannot be spent more than once.

In general, in one aspect, the invention is an electronic cash protocol including the steps of using a one-way function $f_1(x)$ to generate an image $f_1(x_1)$ from a preimage x_1; sending the image $f_1(x_1)$ in an unblinded form to a second party; and receiving from the second party a note including a digital signature. The received signed note represents a commitment by the second party to credit a predetermined amount of money to a first presenter of the preimage x_1 to the second party.

Preferred embodiments include the following features. The electronic cash protocol also includes sending the preimage x_1 to a third party as payment for purchase of goods or services from the third party. Alternatively, it further includes selecting a second preimage x_2; using a second one-way function $f_2(x)$ to generate a second image $f_2(x_2)$ from the second preimage x_2; sending the first preimage x_1 and the unblinded form of the second image $f_2(x_2)$ to the second party; and receiving from the second party a note including a digital signature, the note representing a commitment by the second party to credit the predetermined amount of money to a first presenter of the second preimage x_2 to the second party. In both cases, $f_1(x)$ and $f_2(x)$ are the same function. In the latter case, the sending of the first preimage x_1 and the unblinded form of the second image $f_2(x_2)$ to the second party is performed anonymously and the second party is a bank.

Also in preferred embodiments, the protocol includes the steps of concatenating a signature key of a third party with the first preimage x_1 to form a block of information; encrypting the block of information by using an encryption key of the second party to generate an encrypted block of information; and sending the encrypted block of information to the third party.

In general, in another aspect, the invention is an electronic cash protocol including the steps of receiving a first preimage x_1 from a first party, wherein the preimage x_1 produces a first image $f_1(x_1)$ when processed by a first one-way function $f_1(x)$ and there being associated with said first preimage x_1 a commitment by a second party to credit a predetermined amount ofmoney to a first presenter to the second party of said first preimage x_1; selecting a second preimage x_2 ; using a second one-way function $f_2(x)$ to generate a second image $f_2(x_2)$ from the second preimage x_2; sending the first preimage x_1 and an unblinded form of the second image $f_2(x_2)$ to the second party; and receiving from the second party a note including a digital signature, wherein the note represents a commitment by the second party to credit the predetermined amount of money to a first presenter of the second preimage x_2 to the second party.

In general, in yet another aspect, the invention is an electronic cash protocol including the steps of receiving from a first party an encrypted block of information, wherein the block of encrypted information was generated by first concatenating a public signature key of a second party with a first preimage x_1 to form a block of information and then encrypting the block of information by using an encryption key of a third party; selecting a second preimage x_2; using a second one-way function $f_2(x)$ to generate an image $f(x_2)$ from the preimage x_2; forming a message including the encrypted block of information along with the image $f(x_2)$ in

an unblinded form; sending the message to the third party; and receiving from the third party a signed note including a digital signature, wherein the note represents a commitment by the third party to credit a predetermined amount of money to a first presenter of the preimage x_2 to the third party.

In general, in still another aspect, the invention is an electronic cash protocol including the steps of receiving from a first entity an unblinded form of an image $f_1(x_1)$ that was generated by applying a one-way function $f_1(x)$ to a preimage x_1; generating a message which contains a commitment to credit a predetermined amount of money to a first presenter of the preimage x_1; signing the message with a digital signature; and sending the message along with the digital signature to the first party.

In preferred embodiments, the electronic cash protocol also includes subsequently receiving the preimage x_1 from a third party; checking a database for the preimage x_1; if the preimage x_1 is not found in the database, crediting the third party with the predetermined amount of money; and adding the preimage x_1 to the database. Alternatively, the protocol includes subsequently receiving the preimage x_1 and an unblinded image $f_2(x_2)$ from a third party, wherein the unblinded image $f_2(x_2)$ was generated by applying a one-way function $f_2(x)$ to a preimage x_2; checking a database for the preimage x_1; if the preimage x_1 is not found in the database, generating a signed note including a digital signature, wherein the note represents a commitment to credit the predetermined amount of money to a first presenter of the preimage x_2; and adding the preimage x_1 to the database.

Also in preferred embodiments, the invention features receiving a message from a second party, wherein the message was generated by concatenating an encryption key of a third party with a first preimage x_1 to form a block of information, by encrypting the block of information by using a first encryption key to generate an encrypted first block, and by concatenating an unblinded image $f_2(x_2)$ to the encrypted first block of information, wherein the unblinded image $f_2(x_2)$ was generated by using a one-way function $f_2(x)$ to generate an image $f_2(x_2)$ from a preimage x_2. It further features decrypting the encrypted first block of information; generating a note including a digital signature, wherein the note represents a commitment to credit a predetermined amount of money to a first presenter of the preimage x_2; and sending the note to the second party.

In general, in yet another aspect, the invention is an electronic cash protocol including the steps of sending an unblinded image $f_2(x_2)$ to a second party, wherein the unblinded image $f_2(x_2)$ was generated by applying a one-way function $f_2(x)$ to a preimage x_2; receiving a signed note from the second party, wherein the unblinded note includes a digital signature and represents a commitment to credit the predetermined amount of money to a first presenter of the preimage x_2; and in response to receiving the unblinded note from the second party, authorizing the delivery of goods and/or services to a third party.

The invention offers a simple, inexpensive way of doing cash-like transactions where the item of exchange (i.e., the withdrawn coin) has the properties of actual cash. For example, it is: (1) more or less anonymous; (2) secure; (3) inexpensive to use; and (4) easy to carry around and exchange.

Parties are protected against a dishonest bank's reneging on withdrawn coins by the fact that they keep secret the value x_1 for a particular coin until it is spent. As long as a particular coin $f(x_1)$ is deposited publicly and non-anonymously, the bank can be required to honor it unless it can supply the associated x_1. Of course, the bank can renege on an anonymously exchanged coin $f(x_1)$ during the actual exchange, by claiming upon receiving x_1 that the coin has already been spent. However, the bank cannot possibly know who is being cheated by such a "dine and dash" ploy, and is therefore vulnerable to monitoring and public exposure.

Finally, banks themselves are protected against counterfeiting by the security of the digital signature scheme used to sign electronic coins. Moreover, they are protected against "double-spending" (or "double deposit") by their ability to store x_1 values for coins in perpetuity.

Other advantages and features will become apparent from the following description of the preferred embodiment and from the claims.

BRIEF DESCRIPTION OF THE DRAWING

FIG. **1** is a diagram of a non-anonymous withdrawal protocol;

FIG. **2** is a diagram of an anonymous exchange protocol;

FIG. **3** is a diagram of an anonymous purchase protocol;

FIG. **4** is a diagram of a non-anonymous deposit protocol;

FIG. **5** is a diagram of an anonymous alternate payment protocol;

FIG. **6** is a diagram of an anonymous or non-anonymous "drop" payment or money order protocol; and

FIG. **7** is a diagram of an encrypted money order protocol.

DESCRIPTION OF THE PREFERRED EMBODIMENTS

The ability to communicate anonymously is in some sense necessary a priori if anonymous cash transactions are to occur, since information about a party's communications will obviously reveal information about the party's business dealings. In practice, the anonymity of communication may be based on nothing more than confidence that the telephone company safeguards the confidentiality of its system. Alternatively, parties may place trust in one or more "anonymous remailers" to obscure identities of the parties , or rely on an implementation of one of the other techniques from the publicly available literature.

Suppose, not only that communications between parties are anonymous with respect to third parties, but also that the communicating parties are anonymous to each other. (In typical implementations, the latter condition is a natural consequence of the former, barring self-identification.) A simple, somewhat anonymous electronic cash protocol in such a setting is shown in FIG. **1**.

In the following descriptions of various protocols (see FIGS. 1–7), we generally refer to three parties, namely, a Customer **10**, a Vendor **20**, and a Bank **30**. Customer **10** is of course generally representative of the payor and Vendor **20** is generally representative of the payee. It should be understood, however, that these designations are chosen for purposes of clarity and that they are not meant to limit the scope of the invention. It would be just as valid to have referred to them as Party A, Party B and Party C.

In the figures, the different entities are represented by blocks and the transfers of information from one entity to another are indicated by lines interconnecting the appropriate blocks. Each line represents a transfer of certain information from one entity to another in the direction indicated by an arrow at the end of the line. The information that is transferred is summarized symbolically below the lines.

Though each block is labeled and will be described below as representing a particular entity, it can be implemented by a computing device which performs the computations and the communications that are carried out by that entity. The computing devices might be any of a large variety of electronic devices including, for example, a personal computer, a PCMCIA card, a PDI, a smart-card, a palm-top computer, or a more powerful workstation, just to name a few. The bark side of the protocols that are described below can be implemented by a server programmed to handle electronic transactions, similar to those which currently handle ATM transactions. The server would have multiple telephone lines coming into it and include data storage capability for storing the relevant data.

It should of course also be understood that the computing devices include, either internally or externally, all of the memory that is required for the data and programs that are involved in implementing the protocols. Further more, they include devices (e.g. a modem) which enable them to communicate with other computing devices. In addition, the communications media over which the transfers of information take place can also be any of a large number of possibilities, including telephone lines, cable, the Internet, satellite transmissions, or radio transmissions, for example. In other words, it is not intended that the invention be limited with regard to either the types of devices that are used or the methods of communication that are employed. The possibilities and combinations are limited only by one's imagination.

For the following protocols, it is assumed that Bank **30** chooses and makes publicly available a one-way function f(x). Alternatively, such a function could be made publicly available by any party so long as all parties to the transactions can access and use it. In general, by a one-way function, we mean a function f(x) such that using x_1 produces $f(x_1)$ and given $f(x_1)$ you cannot determine x_1. In the following description, x_1 will also be referred to as a preimage of $f(x_1)$ and $f(x_1)$ will be referred to as an image of x_1.

In practice, perfect one-way functions may not actually exist. That is, for all functions now believed to be one way functions, there may eventually be sufficient computing power or techniques for determining x_1 given $f(x_1)$. Thus, by the phrase one-way function, we mean to also include those functions for which it is very difficult, but not necessarily impossible, to compute x_1 by knowing $f(x_1)$.

The one-way function can be any one of a number of standard hash functions (e.g. MD5, SHA, etc.). In addition, one could use several one-way functions and concatenate them. There are a wide variety of one-way functions known in the art. Typically, many of them are easy to compute, and thus they can be implemented on a smart card.

With that background, the various protocols which embody the invention will now be described, starting with a withdrawal protocol during which a customer obtains "cash" from the bank.

WITHDRAWAL PROTOCOL

A withdrawal is performed in the manner shown in FIG. 1 Customer **10** chooses a random number x_1 and uses f(x) to generate an image of x_1. The value x_1 is a random string obtained from a random number generator to which some post processing may optionally be applied. It may be, for example, 128 bits long. Customer **10** keeps x_1 secret until a payment takes place and then it is sent as the payment.

Customer **10** then withdraws a coin (non-anonymously) from Bank **30** by requesting that Bank **30** associate a monetary value with $f(x_1)$. Bank **30** complies by digitally signing a statement to that effect, thus "certifying" $f(x_1)$ as a valid coin and debits an account which Customer **10** maintains at Bank **30** by the amount of the value of the coin. In other words, Bank **30** issues a statement or representation which indicates in effect that "The first presenter of the preimage of $f(x_1)$ will be credited an amount Z" and then Bank **30** signs or certifies that representation.

Techniques for signing or certifying information (e.g. by using a private key-public key pair) and the use of digital signatures are well known in the art. For further details, refer to any of the widely recognized references in the field, e.g. *Applied Cryptography* by Bruce Schneier, John Wiley & Sons, Inc., (1994).

In general, a signature scheme is a way of tagging a script. It typically uses a public key-private key pair. Public-private keys can be implemented using one-way functions, although a somewhat more practical approach is to use a trap door function, which tends to be more efficient (e.g. see RSA, DSS, ElGamal algorithms described by Schneier). The private key is used to encrypt either the script or a hash of the script to produce a digital signature that is then appended to the script. The digital signature represents a signature of the entity which owns the private key since no other entity can generate such a signature from that script. If a second entity can decrypt the tag using the public key, it knows that the signature was generated by the entity which owns the private key.

Obviously, for certification to work, it is assumed that everyone has and trusts the signing authority's public key and has confidence that the private key has not been compromised.

By publicizing its public key and appending digital signatures to a representation that Bank **30** will pay a specified sum to the entity that first presents a preimage of $f(x_1)$, Bank **30** links itself unambiguously to its commitment, and protects itself against would-be forgers.

The certified representation that is generated by the bank is designated herein as $C(f(x_1))$, also referred to herein as a note. This note is returned to Customer **10**. In addition, the note can be made publicly available since it is of no value to anybody who does not know x_1.

EXCHANGE PROTOCOL

At any time, a party (e.g. Customer **10** or Vendor **20**) can anonymously "exchange" a coin at Bank **30**. Indeed, it is particularly important to do this immediately after receiving a coin from another party to minimize the risk that somebody else will supply x_1 to Bank **30** before the bona fide recipient of the coin. A dishonest party could try to send the coin multiple times by giving x_1 to multiple parties. If that happens, the first recipient to reach Bank **30** will receive its value and all other recipients of the coin will not be able to exchange it for another coin. For Vendor **20**, the timing of the exchange is less crucial because presumably Vendor **20** will not deliver the goods or services that are being purchased until being assured that the coin that was received is still valid.

Referring to FIG. **2**, assuming that Customer **10** wishes to anonymously exchange a coin, Customer **10** supplies to Bank **30** x_1 and another image of the function, $f(x_2)$, for some randomly chosen x_2. In other words, Customer **10** attempts to make a withdrawal as described earlier but simultaneously supplies the amount that is being withdrawn as represented by x_1. Bank **30** simply certifies $f(x_2)$ and keeps x_1 in a database **40** as proof that $f(x_1)$ has already been "spent". It is this exchange that prevents double spending of x_1.

Since $f(x_1)$ and $C(f(x_1))$ are already in the possession of Bank **30**, the sending of that information to Bank **30** along with x_1 and $f(x_2)$ is optional.

If the Bank's side of the protocol is implemented on a server, it automatically stores the x_i's that are received. And each time Bank **30** receives another x_j, the bank first checks it against the x_i's that have already been cashed in (i.e., received).

One can use a series of exchange transactions to obscure who actually is spending the coin. Note that during an exchange transaction, only $f(x_2)$ need be disclosed but not the owner of x_2. Unlike alternative approaches to achieving anonymity, blinding of the coin or other aspects of the transaction is not necessary. Indeed, it is desirable that $f(x_i)$ not be blinded but be made publicly known.

Whatever steps one wishes to take to secure anonymity of communication is sufficient to secure anonymity of the transaction (i.e., achieving anonymity is possible but it is also optional).

This procedure can also be used to make change for a coin of a given value. Instead of sending $f(x_2)$, the party seeking the change can send multiple $f(x)$'s, e.g. $f(x_2)$, (x)'s $f(x_3)$, $f(x_4)$, each for a particular value and the total of which equals the value associated with $f(x_1)$. The bank returns multiple signed notes, $C(f(x_i))$.

Purchase Protocol

Referring to FIG. **3**, the actual spending of coins uses a protocol that is similar to the exchange protocol. The spending party (e.g. Customer **10**) passes x_1 to the receiving party (e.g. Vendor **20**). Since it is likely that Vendor **20** does not have direct and immediate access to $f(x_1)$ and $C(f(x_1))$, Customer **10** also includes this information as part of the transaction. Vendor **20** then immediately calls Bank **30** and exchanges x_1 for a "fresh" coin, assuming that Bank **30** first verifies that it has not previously been spent. Vendor **20** uses the exchange protocol illustrated in FIG. **2** to perform this exchange. Assuming that the exchange was successful, Vendor **20** then delivers to Customer **10** the goods and/or services that were purchased.

DEPOSIT PROTOCOL

Referring to FIG. **4**, unspent coins can also be deposited non-anonymously with Bank **30** at any time. For example, when Vendor **20** wishes to deposit a coin $f(x_1)$ that it has not spent, it sends x_1 to Bank along with a deposit request. Vendor **20** may also optionally send $f(x_1)$ as well as the note $C(f(x_1))$.

Upon receiving x_1 and the deposit request, Bank **30** checks its database to determine whether x_1 has been previously presented to the Bank. Of course, if it had been previously presented, Bank **30** will not credit the Vendor's account and will report to Vendor **20** that it is not a valid coin. If Bank **30** has not previously received x_1, it credits the Vendor's account with the appropriate amount and sends a deposit receipt to Vendor **20** confirming that a credit has been entered.

EXTENSIONS TO THE PROTOCOLS

The exchange payment protocols in the above-described electronic cash scheme permit a number of variations, which can be tailored to the available means of communication and the desired levels of anonymity. For example, referring to FIG. **5**, if Customer **10** has easier access to Bank **30** than Vendor **20**, then Vendor **20** can first supply an $f(x_2)$ to Customer **10**, who then performs the exchange protocol for Vendor **20** and returns the signed coin, i.e., $C(f(x_2))$, as proof of payment. As mentioned previously, the exchange protocol may be performed anonymously.

Alternatively, if both Customer **10** and Vendor **20** have better communications access to Bank **30** than to each other, then the parties may use a "drop" payment protocol, such as is illustrated in FIG. **6**. In accordance with this protocol, Customer **10** drops off the payment at Bank **30** for Vendor **20** and Vendor **20** subsequently picks up the payment from Bank **30**.

The steps of the "drop" payment protocol are as follows. First, Customer **10** supplies an x_2 for a valid coin of a specific amount to Bank **30**, along with a public signature key p of Vendor **20**, and other information relating to the transaction. For example, among the other information Customer **10** might wish to identify the goods being purchased, to identify the transaction and/or the vendor, and to indicate the declared of the customer intentions regarding payment, thereby essentially turning the cash into a kind of "electronic money order". Optionally, Customer **10** can also send $f(x_1)$ and the note $C(f(x_1))$, but as pointed out earlier, since this information is already available to Bank **30**, sending it may not be necessary.

Note that the a record that may be assembled from the other information supplied by Customer **10** may be of particular use in remote payment settings, where the nature of the transaction is not otherwise implicit in the action of payment itself, as is typically the case for in-person payments.

If Vendor **20** does not wish to remain anonymous, the public signature key may be publicly associated with the identity of Vendor **20**; or if anonymity is desired, the public signature key can be a special-purpose public signature key with no associated identity. In the latter case, the public key is passed confidentially to trusted acquaintances or simply publicized under a pseudonym.

Bank **30** agrees to assign the amount associated with x_1 to the first coin $f(x_i)$ presented to it that it is also signed using the private signature key that corresponds with the previously-delivered public signature key p. Thus to obtain the payment for the goods that Customer **10** wishes to purchase, Vendor **20** simply makes a withdrawal from Bank **30** using the protocol previously described in connection with FIG. **1**. That is, Vendor **20** randomly selects an x_2, and uses $f(x)$ to generate its image $f(x_2)$. In this instance, however, Vendor **20** signs $f(x_2)$ with its private signature key before sending $f(x_2)$ to Bank **30**. In addition, in this case the withdrawal is not from the account of the vendor but is simply a transfer of the amount previously supplied by Customer **10**.

Bank **30** uses the Vendor's public signature key of the vendor to verify that $f(x_2)$ is signed by Vendor **20** (i.e., by the party to whom the money transfer is to be made). Upon confirming the signature on $f(x_2)$, Bank **30** issues a note $C(f(x_2))$ which it sends to Vendor **20**.

After Vendor **20** receives the note $C(f(x_2))$ confirming that the money has been received, Vendor **20** sends the goods to Customer **10**. of course, theoretically Bank **30** could cheat by simply keeping the money instead of assigning it to the payee. However, we rely on the anonymity of the payer or at least the possibility that the payer may be exposing the transaction to public monitoring to keep Bank **30** honest.

In a setting where communications among the parties may be intercepted, there are a number of ways of securing the exchange protocols and, in particular, the secret x value

passed therein from eavesdroppers. The most natural method is public key encryption. If parties know public encryption keys of each other, as well as of the bank, then all of the above protocols can function equally well in the eavesdropper-threatened setting, as long as every message, except those sent by Bank **30**, is encrypted using the public encryption key of the receiver or using a symmetric "session key" encrypted using the receiver's public encryption key. The messages of the bank, of course, can be considered non-confidential, since they consist only of signed coins of the form $f(x_i)$, with x_i kept secret by someone else. The use of message authentication codes, or MAC's, with each encryption makes it possible also to ensure that the message is not even tampered with by someone other than the sender before arriving at its destination.

The use of public-key encryption also makes possible another kind of "electronic money order." In this case, which is illustrated in FIG. **7** and referred to generally as an encrypted money order protocol, Customer **10** encrypts the secret x_i value for some valid electronic coin, along with the public key p of Vendor **20** and any other desired identity or transaction information, as in the case of the previous "drop" protocol. Customer **10** encrypts this information by using the public encryption key of the bank or by using a session key which is then encrypted using the public encryption key of the bank. Customer **10** then sends the encrypted data directly to Vendor **20**.

To "cash" it, Vendor **20** selects a random value x_2, generates its image $f(x_2)$ and appends $f(x_2)$ to the message E that was received from Customer **10**. As before, $f(x_2)$ is to be signed by the bank so that it will represent the transfer of cash to Vendor **20**. Vendor **20** signs the complete message (or at least $f(x_2)$) using the private signature key associated with public signature p, and passes E, $f(x_2)$ and the signature to Bank **30**. Optionally, Vendor **20** may further encrypt this message in the manner described before, i.e., using the banks encryption key and possibly an additional symmetric key.

After Bank **30** has decrypted the message from Vendor **20** by using its private key, it then checks its database to determine if it does not already have x_1 stored therein, and if it is not found, Bank **30** stores x_1. Bank **30** then generates a note $C(f(x_2)$ representing a cash transfer to Vendor **20** in the amount of the value associated with $f(x_1)$. The note is then sent to Vendor **20** which after receipt and verification sends the purchased goods to Customer **10**.

In effect, this encrypted last protocol is identical to the previous one. The addition of encryption has simply permitted the payer to pass the "money order" via the payee, while ensuring that the secret and additional information provided by the payer is not tampered with.

It may be beneficial for the note, $C(f(x_i)$, to include an expiration date. In that case, the stored x_is in the database of Bank **30** will not grow too large. That is, x_1s will not have to kept around in the database of the Bank forever. To prevent the value of the coins from expiring, the smartcard (or whatever equipment is handling the customer's transactions) could automatically exchange the old coins for new ones with a new expiration date.

The expiration date also makes the money refundable. If a user's smartcard breaks and all of the x_i s are lost, the user can simply present the $f(x_i)$ to Bank **30** with the request that if they are not claimed within 3 months after the expiration date, then the user, e.g. customer **10** wants to be credited with the amount of the value of the coins. For this to work, however, during the original communication with Bank **30** at which time the coins are withdrawn, Customer **10** must identify himself or herself.

The customer side of the protocols can be easily implemented using a smartcard since only the x_i's need to be stored and they typically will not require a lot of memory. To prevent theft of the x_i's by somebody who would steal the smartcard, a PIN can be used in the smartcard which is secret and which must be entered by the user before any of the x_i's can be accessed.

Note that it was also assumed that all of the interactions that were described above were automated. That is, they were automatically carried out by an appropriately programmed computer or processor that was under the control of the party for whom the transaction is being implemented.

Other embodiments are within the following claims. For example, another way to link identifying information to electronic coins is to use the secret value x_i to perform the linking. In the above-description, it was assumed that the secret values x_i are generated randomly by the coin's creator. The secret value can, however, be generated as the image of some identifying data under a one-way function h(x), which could perhaps be the same function f(x) that is used in the construction of normal electronic coins. The identifying information might include the purpose and date of the payment and the name of the payer and the intended payee —in short, all of the information that the payer might have wished the bank to archive. This information would then be passed through h(x) to generate an x_i, which serves as the secret value.

In this case, there would be no need for the bank to archive transaction information received with electronic payments in either the "drop" or "electronic money order" protocols described above. In fact, all that is required is that the payment be labeled as requiring that the payee be non-anonymous. As long as the bank positively identifies the payee and keeps normal records of the transaction, including the payee's identity, the payer can later demonstrate "payership" by publicly revealing the pre-image of x_i; under f(x), which as indicated above might include information regarding the purpose and date of the payment and the names of the payer and the intended payee. The payer can obtain coins with such implicitly information carrying x_i; values simply by constructing them normally, and then exchanging other coins for them. In this context, however, the payment information need not even be sent to the bank, since it is implicitly contained in x_i. Hence, the only information that the payer needs to pass securely to the bank is the public signature key to be used to identify the payee, which implicitly communicates the requirements that the payer be non-anonymous.

In fact, even this latter requirement of signature-based payee identification can be eliminated if information is embedded in x_i(or $f(x_i)$ to the effect that the bank is not to honor the cash non-anonymously. For example, some property of x_i (e.g. the value of the first bit being 1) might be publicly declared by the bank to indicate that the coin in question will only be honored non-anonymously. A payer can then generate a secret x_i by computing $f(s_j)$, where s_j is the concatenation of the payment information for an intended transaction with some random value r, chosen such that x_i has the non-anonymity property. Note that the property should be chosen such that roughly half the pre-images s_j of f(s) of any particular length result in an $f(s_j)$ with the property, so that not many attempts to choose r will be necessary before one is found that has the desired effect on x_i). This coin would now have the property that anyone

presenting it for redemption must also provide an identity and prove it to the bank's satisfaction so that the bank can record the identity of the exchanger as part of its normal accounting. As a result, the coin's creator would later be able to demonstrate its origin, as well as other details of the transaction in which it was intended to be used, by referring to the bank's accounting records and revealing the s_j, used to generate x_j. Hence, even if the coin is spent completely normally, with no extra encryption or attendant information for the bank, it still provides the payer with all the protection furnished by the "electronic money order" described earlier.

What is claimed is:

1. A method of implementing an electronic cash protocol comprising the steps of:

using a one-way function $f_1(x)$ to generate an image $f_1(x_1)$ from a preimage x_1;

sending the image $f_1(x_1)$ in an unblinded form to a second party; and

receiving from the second party a note including a digital signature, said note representing a commitment by the second party to credit a predetermined amount of money to a first presenter of said preimage x_1 to the second party.

2. The method of claim 1 further comprising sending the preimage x_1 to a third party as payment for purchase of goods or services from the third party.

3. The method of claim 1 further comprising:

selecting a second preimage x_2;

using a second one-way function $f_2(x)$to generate a second image $f_2(x_2)$ from the second preimage x_2;

sending the first preimage x_1 and the unblinded form of the second image $f_2(x_2)$ to the second party; and

receiving from the second party a second note including a digital signature, said second note representing a commitment by the second party to credit said predetermined amount of money to a first presenter of said second preimage x_2 to the second party.

4. The method of claim 3 wherein functions $f_1(x)$ and $f_2(x)$ are identical to each other.

5. The method of claim 4 wherein the step of sending the first preimage x_1 and the unblinded form of the second image $f_2(x_2)$ to the second party is performed anonymously.

6. The method of claim 5 wherein the second party is a bank.

7. The method of claim 3 further comprising sending the second preimage x_2 to a third party as payment for purchase of goods or services from the third party.

8. The method of claim 1 further comprising:

concatenating a signature key of a third party with the first preimage x_1 to form a block of information;

encrypting the block of information by using an encryption key of the second party to generate an encrypted block of information; and

sending the encrypted block of information to the third party.

9. A method of implementing an electronic cash protocol comprising the steps of:

receiving a first preimage x_1 from a first party, said preimage x_1 producing a first image $f_1(x_1)$ when processed by a first one-way function $f_1(x)$ and there being associated with said first preimage x_1 a commitment by a second party to credit a predetermined amount of money to a first presenter to the second party of said first preimage x_1;

selecting a second preimage x_2;

using a second one-way function $f_2(x)$to generate a second image $f_2(x_2)$ from the second preimage x_2;

sending the first preimage x_1 and an unblinded form of the second image $f_2(x_2)$ to the second party; and

receiving from the second party a note including a digital signature, said note representing a commitment by the second party to credit said predetermined amount of money to a first presenter of said second preimage x_2 to the second party.

10. The method of claim 9 wherein functions $f_1(x)$ and $f_2(x)$ are identical to each other.

11. The method of claim 9 wherein the step of sending the first preimage x_1 and the unblinded form of the second image $f_2(x_2)$ to the second party is performed anonymously.

12. A method of implementing an electronic cash protocol comprising the steps of:

receiving from a first party an encrypted block of information, wherein said block of encrypted information was generated by first concatenating a public signature key of a second party with a first preimage x_1 to form a block of information and then encrypting the block of information by using an encryption key of a third party;

selecting a second preimage x_2;

using a second one-way function $f_2(x)$ to generate an image $f(x_2)$ from the preimage x_2;

forming a message including the encrypted block of information along with the image $f(x_2)$ in an unblinded form;

sending the message to the third party; and

receiving from the third party a signed note including a digital signature, said note representing a commitment by the third party to credit a predetermined amount of money to a first presenter of said preimage x_2 to the third party.

13. The method of claim 12 wherein functions $f_1(x)$ and $f_2(x)$ are identical to each other.

14. The method of claim 12 further comprising signing the message before sending it to the third party, wherein the step of signing is performed using a private signature key corresponding to a public signature key possessed by the third party.

15. The method of claim 12 wherein the second party is the party receiving the encrypted block of information from the first party.

16. A method of implementing an electronic cash protocol comprising the steps of:

receiving from a first entity an unblinded form of an image $f_1(x_1)$ that was generated by applying a one-way function $f_1(x)$ to a preimage x_1;

generating a message which contains a commitment to credit a predetermined amount of money to a first presenter of said preimage x_1;

signing said message with a digital signature; and

sending said message along with said digital signature to said first party.

17. The method of claim 16 wherein the receiving party maintains an account for the first entity and wherein said protocol further comprises debiting the first party's account by the predetermined amount of money.

18. The method of claim 16 further comprising:

subsequently receiving the preimage x_1 from a third party;

checking a database for the preimage x_1;

if the preimage x_1 is not found in said database, crediting the third party with said predetermined amount of money; and

adding the preimage x_1 to said database.

19. The method of claim 16 further comprising:

subsequently receiving the preimage x_1 and an unblinded image $f_2(x_2)$ from a third party, wherein the unblinded image $f_2(x_2)$ was generated by applying a one-way function $f_2(x)$ to a preimage x_2;

checking a database for the preimage x_1;

if the preimage x_1 is not found in said database, generating a signed note including a digital signature, said note representing a commitment to credit said predetermined amount of money to a first presenter of said preimage x_2; and

adding the preimage x_1 to said database.

20. The method of claim 19 wherein functions $f_1(x)$ and $f_2(x)$ are identical to each other.

21. The method of claim 16 further comprising:

receiving a message from a second party, wherein said message was generated by concatenating an encryption key of a third party with a first preimage x_1 to form a block of information, by encrypting the block of information by using a first encryption key to generate an encrypted first block, and by concatenating an unblinded image $f_2(x_2)$ to the encrypted first block of information, wherein said unblinded image $f_2(x_2)$ was generated by using a one-way function $f_2(x)$ to generate an image $f_2(x_2)$ from a preimage x_2;

decrypting the encrypted first block of information;

generating a note including a digital signature, said note representing a commitment to credit a predetermined amount of money to a first presenter of said preimage x_2; and

sending said note to the second party.

22. The method of claim 21 wherein functions $f_1(x)$ and $f_2(x)$ are identical to each other.

23. The method of claim 21 further comprising:

checking a database for the preimage x_1;

generating the signed note only if the preimage x_1 is not found in said database; and

adding the preimage x_1 to said database.

24. A method of implementing an electronic cash protocol comprising the steps of:

obtaining a first image $f(x_1)$ and a first preimage x_1, wherein said first preimage x_1 has a predetermined monetary value associated therewith;

selecting a second preimage x_2;

using a second one-way function $f_2(x)$to generate a second image $f_2(x_2)$ from the second preimage x_2;

sending the first preimage x_1 and an unblinded form of the second image $f_2(x_2)$ to the second party; and

receiving from the second party a note including a digital signature, said note representing a commitment by the second party to credit a predetermined amount of money to a first presenter of said second preimage x_2 to the second party, wherein said predetermined amount of money is no greater than said predetermined monetary value.

25. The method of claim 24 wherein said predetermined amount of money is less than said predetermined monetary value.

26. The method of claim 24 wherein $fx_1(x)$ equals $f_2(x)$.

27. A method of implementing an electronic cash protocol comprising the steps of:

obtaining a first image $f(x_1)$ and a first preimage x_1, wherein said first preimage x_1 has a predetermined monetary value associated therewith;

selecting a plurality of preimages x_i, wherein I is an integer index that runs from 1 to n, where n is a positive integer;

using a second one-way function $f_2(x)$ to generate a plurality of images $f_2(x_i)$ from the second preimages x_i;

sending the first preimage x_1 and an unblinded form of all of the images $f_2(x_i)$ to the second party; and

receiving from the second party a plurality of each including a digital signature, said plurality of notes equal in number to the plurality of images $f_2(x_i)$ and representing a plurality of predetermined amounts, each of said plurality of notes representing a commitment by the second party to credit a corresponding different one of said plurality of predetermined amounts of money to a first presenter of the corresponding preimage x_i to the second party, wherein the total of said plurality of predetermined amounts of money equals said predetermined monetary value.

28. A method of implementing an electronic cash protocol comprising the steps of:

obtaining a first image $f(x_1)$ and a first preimage x_1, wherein said first preimage x_1 has a predetermined monetary value associated therewith;

concatenating a signature key of a second party with the first preimage x_1 to form a block of information;

encrypting the block of information by using an encryption key of a third party to generate an encrypted block of information; and

sending the encrypted block of information to the third party.

29. The method of claim 28 further comprising concatenating other information with the signature key of a second party and the first preimage x_1 to form the block of information.

30. A method of implementing an electronic cash protocol comprising the steps of:

sending an unblinded image $f_2(x_2)$ to a second party, wherein the unblinded image $f_2(x_2)$ was generated by applying a one-way function $f_2(x)$ to a preimage x_2;

receiving a signed note from the second party, said unblinded note including a digital signature, said unblinded note representing a commitment to credit said predetermined amount of money to a first presenter of said preimage x_2; and

in response to receiving the unblinded note from the second party, authorizing the delivery of goods or services to a third party.

* * * * *

UNITED STATES PATENT AND TRADEMARK OFFICE

CERTIFICATE OF CORRECTION

PATENT NO. : 5,768,385 Page 1 of 2

DATED : June 16, 1998

INVENTOR(S) : Daniel R. Simon

It is certified that error appears in the above-identified patent and that said Letters Patent is hereby corrected as shown below:

Column 1, line 36 Change "preserved." to --preserved?--;

Column 2, line 1 Delete "and";

Column 2, line 4 Change "The anonymity of spenders" to --Not only does the system protect the anonymity of spenders--;

Column 3, line 50 Change "f_2 (X_2)" to --$f_2(X_2)$--;

Column 5, line 9 Change "bark" to --bank's--;

Column 8, line 16 After "information", insert a comma;

Column 8, line 19 Change "the declared of the customer intentions" to --the declared intentions of the customer--;

Column 8, line 42 After "Thus", insert a comma;

Column 8, line 53 Delete "Vendor's";

UNITED STATES PATENT AND TRADEMARK OFFICE

CERTIFICATE OF CORRECTION

PATENT NO. : 5,768,385 Page 2 of 2

DATED : June 16, 1998

INVENTOR(S) : Daniel R. Simon

It is certified that error appears in the above-identified patent and that said Letters Patent is hereby corrected as shown below:

Column 9, line 42 After "and", insert a comma;

Column 9, line 56 After "to", insert --be--;

Column 10, line 5 After "stored", insert a semicolon;

Column 10, line 5 Delete "and".

Signed and Sealed this

Twenty-second Day of September, 1998

Attest:

BRUCE LEHMAN

Attesting Officer *Commissioner of Patents and Trademarks*

US005930777A

United States Patent [19]

Barber

[11] Patent Number: 5,930,777

[45] Date of Patent: Jul. 27, 1999

[54] **METHOD OF CHARGING FOR PAY-PER-ACCESS INFORMATION OVER A NETWORK**

[76] Inventor: **Timothy P. Barber**, 11931 Chalon La., San Diego, Calif. 92128

[21] Appl. No.: **08/862,496**

[22] Filed: **May 23, 1997**

Related U.S. Application Data

[60] Provisional application No. 60/043,020, Apr. 15, 1997.

[51] **Int. Cl.**[6] **G06F 17/60**

[52] **U.S. Cl.** **705/40**; 705/26; 705/27; 705/17; 380/24; 380/25; 395/200.33

[58] **Field of Search** 705/40, 26, 27, 705/41, 16, 17, 44; 380/24, 25; 395/200.33, 200.59

[56] **References Cited**

U.S. PATENT DOCUMENTS

5,692,132	11/1997	Hogan	705/27
5,708,780	1/1998	Levergood et al.	395/200.12
5,715,314	2/1998	Payne et al.	380/24
5,724,424	3/1998	Gifford	380/24
5,729,594	3/1998	Klingman	379/93.12
5,802,497	9/1998	Manasse	705/27

OTHER PUBLICATIONS

Rihaczek, Karl; "TeleTrusT–OSIS and Communication Security"; Computers and Security; vol. 6, No. 3, pp. 206–218, Jun. 1987.

Rihaczek, Karl; "Teletrust"; Computer Networks and ISDN systems; vol. 13, No. 3; pp. 235–239, 1987.

Hallam–Baker, Phillip M.; "Micro Payment Transfer Protocol (MPTP) Version 0.1" Internet Draft; pp. 1–28, Nov. 1995.

Manasse, Mark S.; "The Millicent protocols for electonic commerce"; Usenix Association; pp. 117–123, Jul. 1995.

Akashi, Osamu; Moriyasu, Kenji; and Terauchi, Atsushi; "Information Distribution by FleaMarket System" IEEE pp. 139–146, 1996.

"SubScrip—An efficient protocol for pay–per–view payments on the Internet," Andreas Furche & Graham Wrightson, Dept. of Computer, Science, U. of Newcastle, Oct. 16, 1996, pp. 1–5.

"PayWord and MicroMint: Two simple micropayment schemes," Ronald L. Rivest* and Adi Shamir**, *MIT Laboratory for Computer Science, **Weizmann Institute of Science, May 7, 1996, pp. 1–18.

"iKP—A Family of Secure Electronic Payment Protocols," IBM Research, Mar. 15, 1995 pp. 1–17.

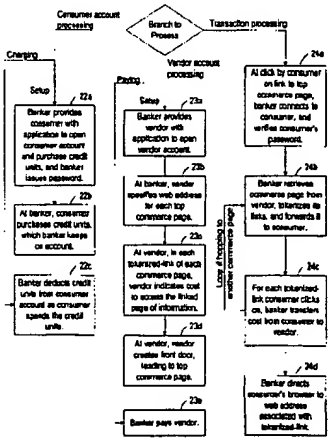
(List continued on next page.)

Primary Examiner—James P. Trammell
Assistant Examiner—Demetra R. Smith
Attorney, Agent, or Firm—Ware, Fressola, Van Der Sluys & Adolphson LLP

[57] **ABSTRACT**

A method for charging a consumer for access, over a network, to a vendor's information; in particular, a method for this pay-per-access over the Internet. The method uses a third-party, called a banker, to mint tokens identified with particular information a consumer might want to purchase. The tokens are immediately available to the consumer because of the consumer's having already established an account with the banker, and purchased what are here called credit units, which can have a value of only a fraction of a cent, allowing vendors to charge very little for access to their information. A token is pre-authorization for a consumer to pay for access for a particular page of information. In one embodiment, when a consumer makes a purchase, i.e. chooses to access a Web page for which a vendor makes a charge, the transaction is routed through the banker, which charges in credit units (those already on account), and credits the vendor account. The vendor later redeems for payment whatever credit units have been credited to the vendor's banker account, not necessarily only those credit units resulting from transactions with a particular consumer. In another embodiment, a vendor uses a franker to test if a token is valid; this enables the vendor to collect a token from a consumer and later redeem it for payment with a banker. The method also allows for a vendor and consumer to have accounts with different bankers.

5 Claims, 5 Drawing Sheets



OTHER PUBLICATIONS

"Mini-Pay: Charging per Clickon the Web," IBM Research-Haifa Research Lab-Tel-Aviv Annex Apr. 10, 1997, pp. 1-20.

"Millicent: Frequently Asked Questions," Apr. 15, 1997, pp. 1-3.

"Millicent-specific elements for an HTTP payment protocol," Apr. 15, 1997, pp. 1-8.

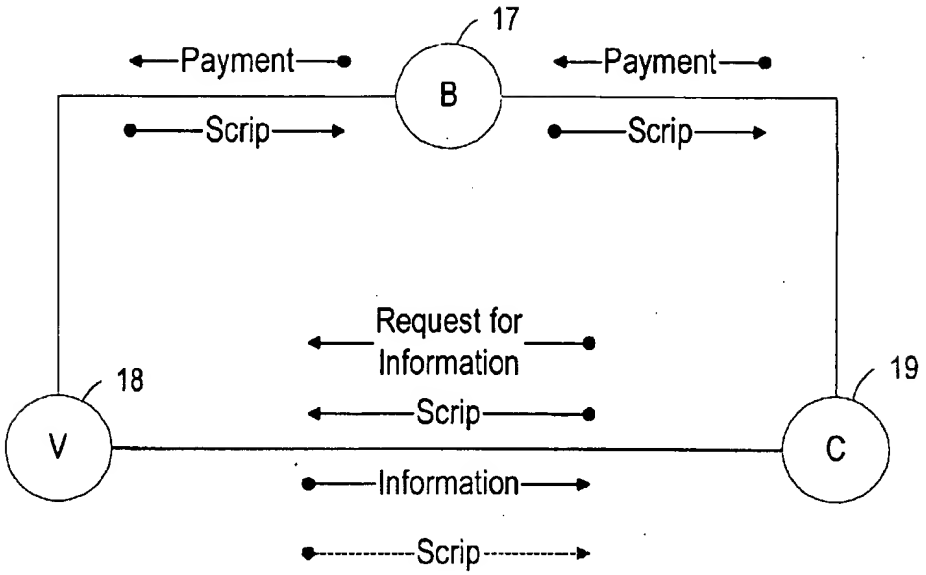


Fig. 1a (Prior Art)

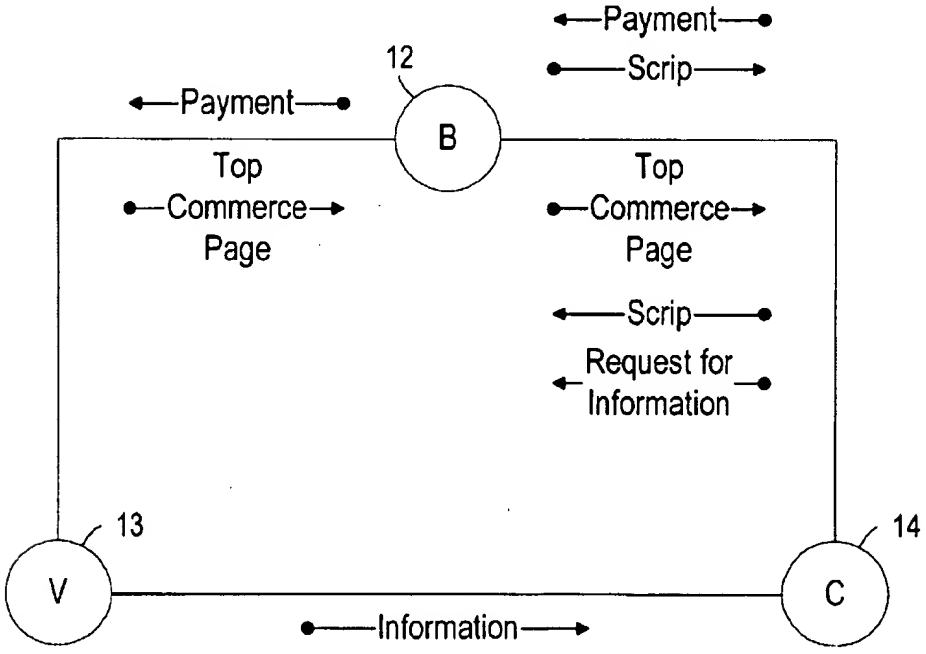


Fig. 1b

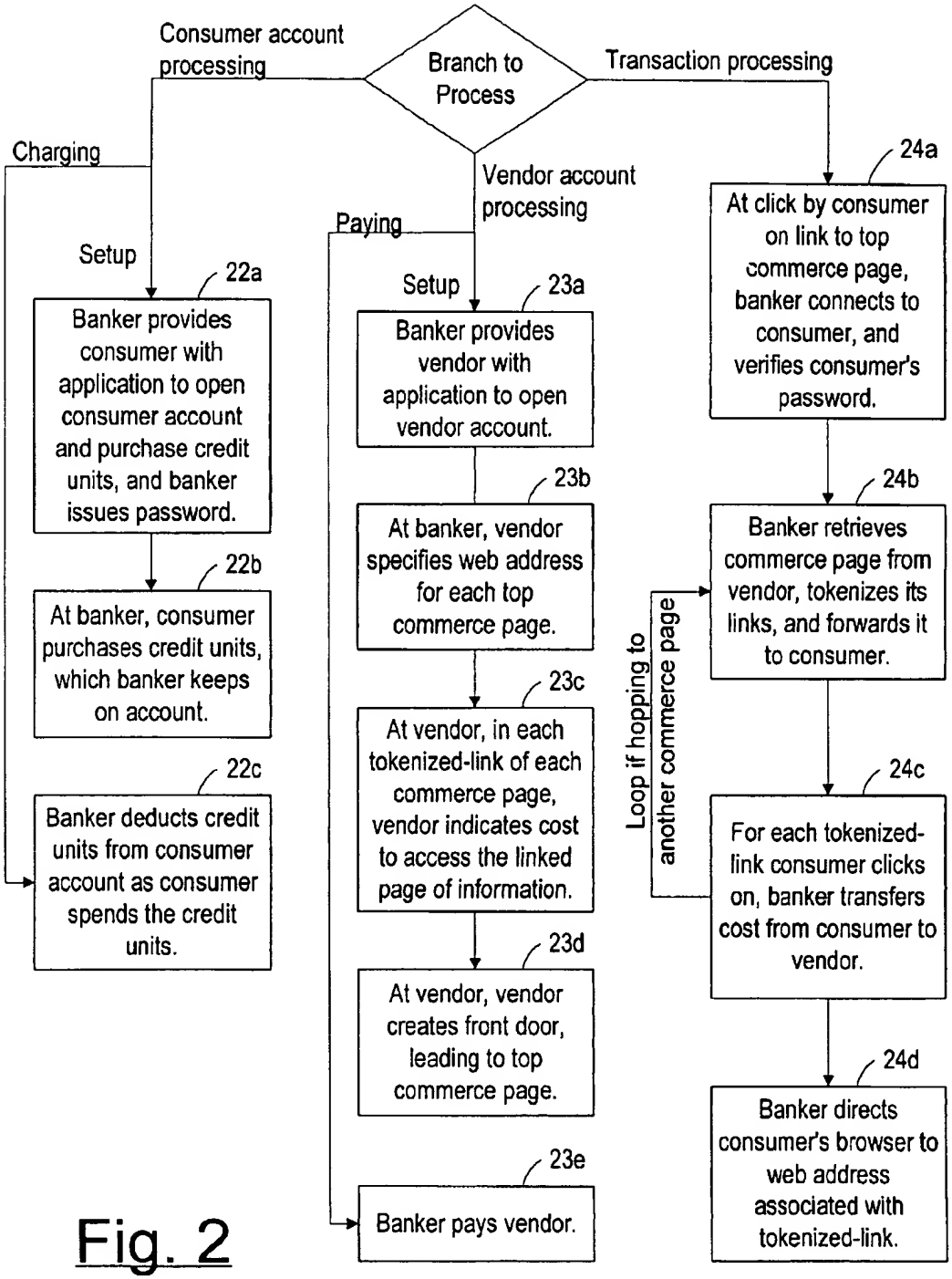

Branch to Process
Consumer account processing
Transaction processing
Charging
Vendor account processing
Paying
Setup
22a
Banker provides consumer with application to open consumer account and purchase credit units, and banker issues password.
22b
At banker, consumer purchases credit units, which banker keeps on account.
22c
Banker deducts credit units from consumer account as consumer spends the credit units.
Setup
23a
Banker provides vendor with application to open vendor account.
23b
At banker, vendor specifies web address for each top commerce page.
23c
At vendor, in each tokenized-link of each commerce page, vendor indicates cost to access the linked page of information.
23d
At vendor, vendor creates front door, leading to top commerce page.
23e
Banker pays vendor.
24a
At click by consumer on link to top commerce page, banker connects to consumer, and verifies consumer's password.
24b
Banker retrieves commerce page from vendor, tokenizes its links, and forwards it to consumer.
Loop if hopping to another commerce page
24c
For each tokenized-link consumer clicks on, banker transfers cost from consumer to vendor.
24d
Banker directs consumer's browser to web address associated with tokenized-link.

Fig. 2

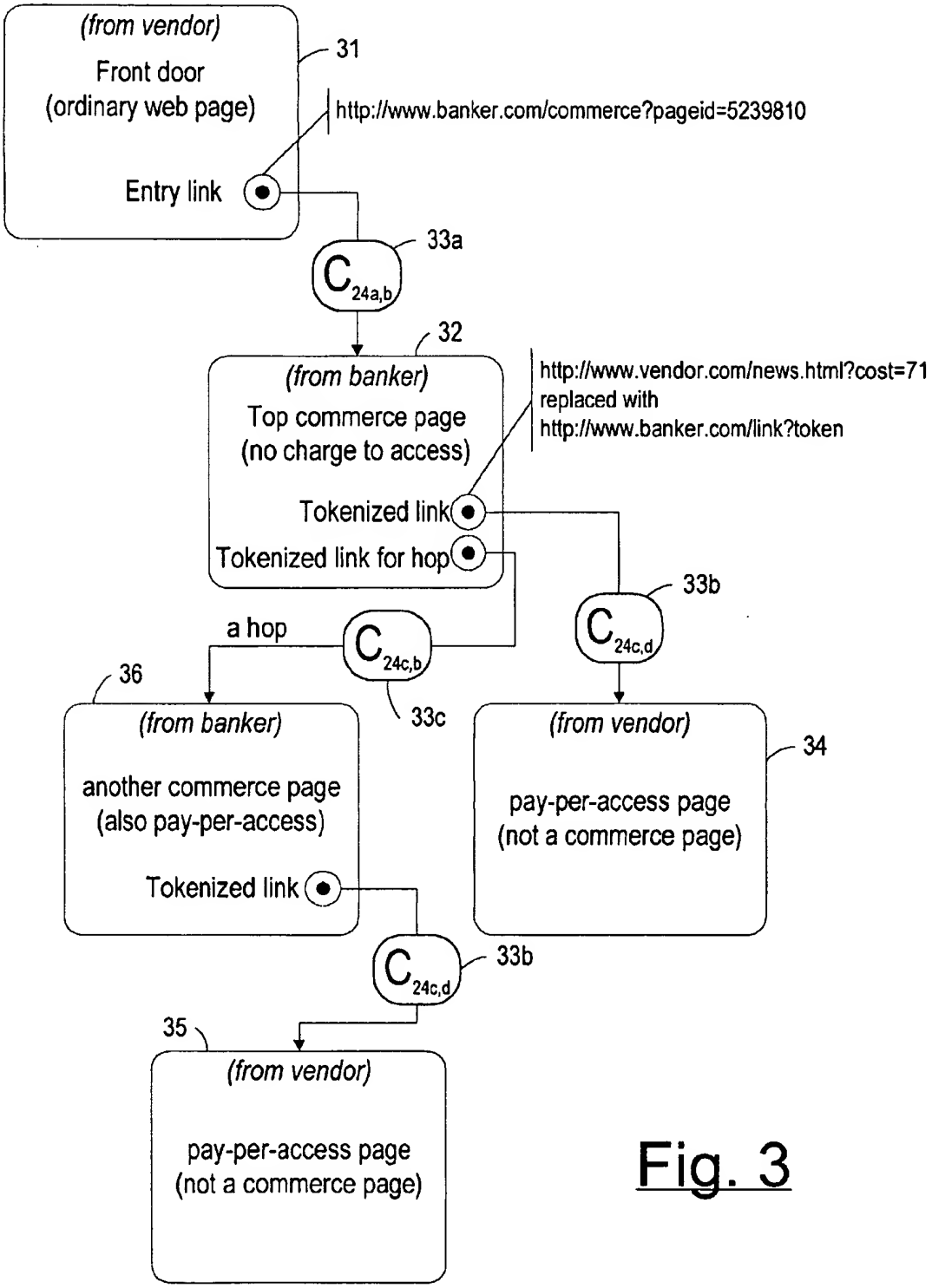

(from vendor)
Front door
(ordinary web page)
31
http://www.banker.com/commerce?pageid=5239810
Entry link
33a
C24a,b
32
(from banker)
Top commerce page
(no charge to access)
http://www.vendor.com/news.html?cost=71
replaced with
http://www.banker.com/link?token
Tokenized link
Tokenized link for hop
33b
C24c,d
a hop
C24c,b
33c
36
(from banker)
another commerce page
(also pay-per-access)
Tokenized link
(from vendor)
pay-per-access page
(not a commerce page)
34
C24c,d
33b
35
(from vendor)
pay-per-access page
(not a commerce page)

Fig. 3

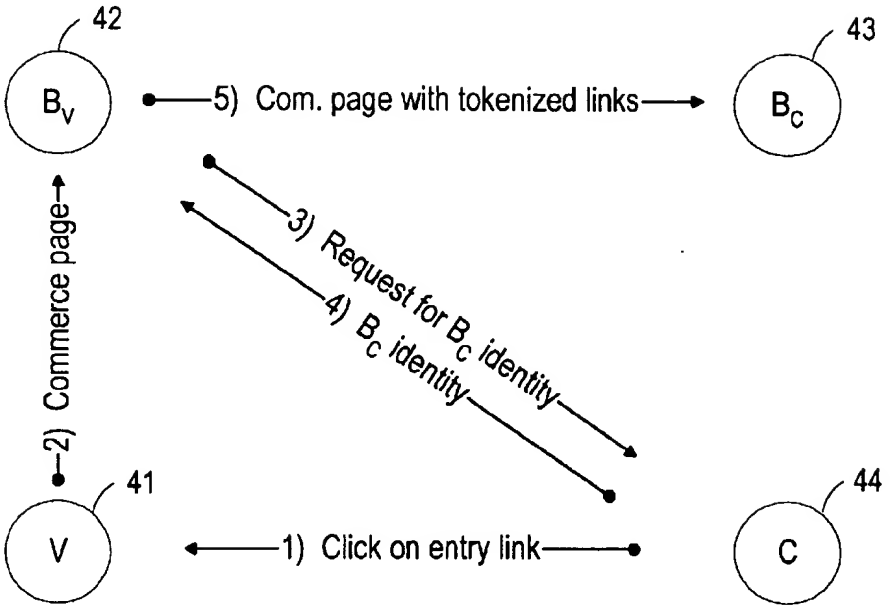

42
43
B_V
5) Com. page with tokenized links
B_C
3) Request for B_C identity
4) B_C identity
2) Commerce page
41
44
V
1) Click on entry link
C

Fig. 4a

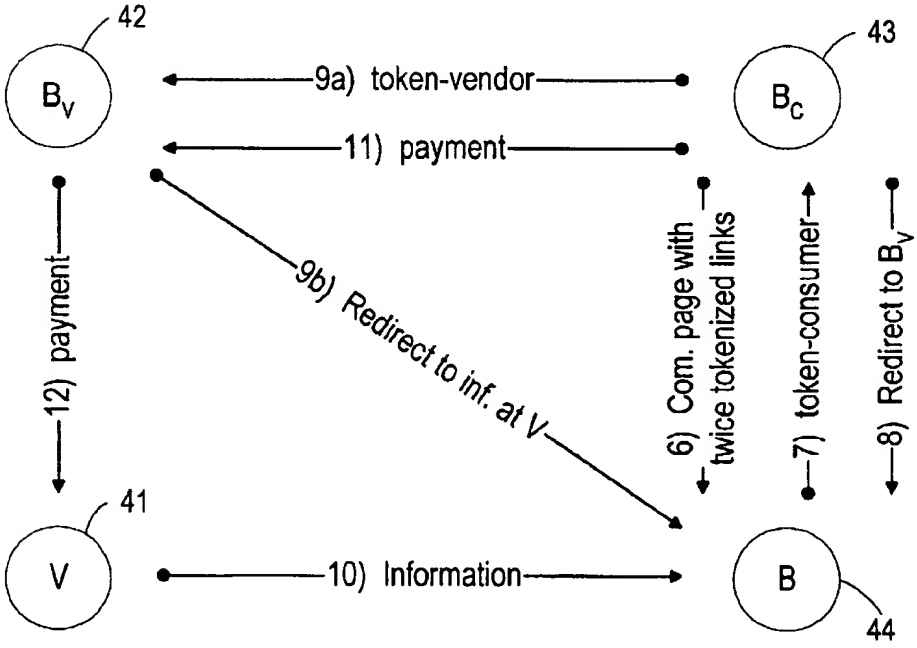

42
43
B_V
9a) token-vendor
B_C
11) payment
9b) Redirect to inf. at V
6) Com. page with twice tokenized links
7) token-consumer
8) Redirect to B_V
12) payment
41
V
10) Information
B
44

Fig. 4b

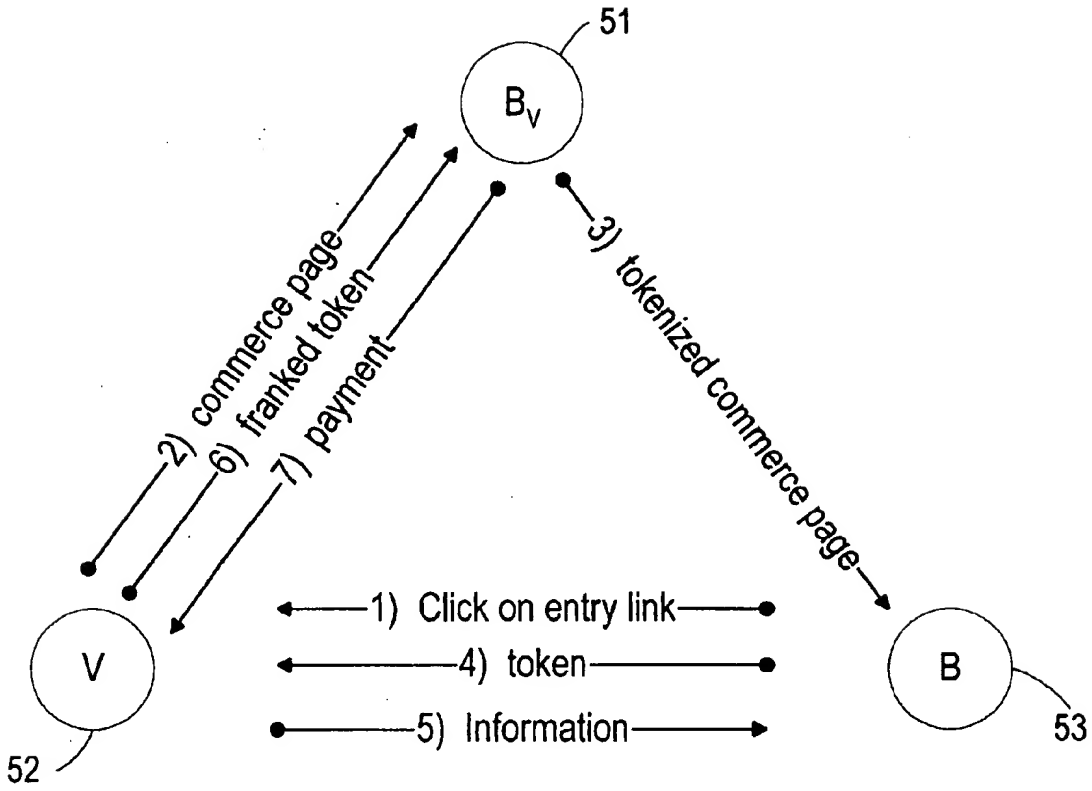

51
B_V
2) commerce page
6) franked token
7) payment
3) tokenized commerce page
1) Click on entry link
4) token
5) Information
V
B
52
53

Fig. 5

METHOD OF CHARGING FOR PAY-PER-ACCESS INFORMATION OVER A NETWORK

CROSS REFERENCE TO RELATED APPLICATION

Reference is made to and priority claimed from U.S. provisional application Ser. No. 60/043,020, filed Apr. 15, 1997, entitled INTERNET PAYLINK TRANSFER METHOD.

BACKGROUND OF THE INVENTION

1. Technical Field

The present invention pertains to the field of charging a fee for information provided over a network. More particularly, the present invention pertains to a method of handling access to information over the Internet in a way that makes feasible a quite small charge per access, and does not require the consumer to have an account with the information provider.

2. Description of Related Art

Many information vendors on the Internet are accessed by consumers only once or twice for only small amounts of particular information at a time. To operate competitively, the information vendor must charge the consumer only a few cents for each access. Because of the small charge, the overhead in charging a consumer must be kept under tight control. In addition, the method used to charge for the access must verify that the consumer will pay. Early solutions to these problems relied on using a central authority to verify access, slowing down the transaction. The challenge of keeping overhead low and providing rapid access persists.

There are essentially two approaches to the pay-per-access problem: token-based and account-based. Protocols are built up around methods based on each of these approaches; a protocol, as used here, is a specific implementation of a method of charging for a consumer's access to a vendor's information.

In general, token-based methods have a consumer purchase electronic tokens from a bank. To access a vendor's information, the consumer will pay the vendor using the tokens. The vendor can then go to the bank and deposit the tokens or redeem them for money. An account-based method works like a charge-card method of paying for merchandise. A consumer authorizes a bank to transfer funds from the consumer's account to a vendor's account in exchange for receiving information from the vendor. The funds transfer is performed by the bank.

Token-based methods, compared to account-based methods, are generally considered to have the potential to achieve lower transaction costs, but do not by themselves meet the challenge of providing rapid on-line clearance. The account-based methods certainly meet the challenge of providing on-line clearance, but these kinds of methods often tend to fall short in meeting the need for rapid access at low cost.

Protocols have been proposed by credit card companies that essentially model the credit card system on the Internet. Because these methods use accounts identified by consumer name, they fail in providing another desirable characteristic: consumer anonymity. On the other hand, token-based methods have been developed that do provide consumer anonymity. However, in both the token-based methods and account-based methods developed so far, overhead is usually too high to process transactions for which only a small

charge is made, and all the methods require on-line verification, which unavoidably slows a transaction.

Much of the prior art, whether for an account-based or token-based method, can be understood in terms of FIG. 1*a*. In either kind of method, a consumer **19** first makes a payment to a broker **17** in exchange for scrip. The term scrip is used here in a generic sense to indicate a data object used in place of money. As such, the term encompasses both tokens and authorization to charge an account. The term payment is used here to indicate conveyance by a payer of money, as opposed to scrip, such as by a personal check, or authorization to charge the payer money, such as by charging a credit card owned by the payer.

With the scrip purchased from the broker **17**, the consumer can purchase information from an information vendor **18**. To do so, the consumer requests the information he wants using the vendor's interface, which usually indicates to the consumer the cost of the information. The vendor's interface to the consumer will take scrip for the amount charged for the information, and send the consumer the information. Finally, the vendor will redeem for payment from the broker the scrip the vendor has collected. The vendor may of course make the redemption at a later time so as to redeem scrip from multiple users.

This model is easily applied to token-based methods, and in these methods there is sometimes one additional exchange: the vendor may make change, returning scrip to the consumer equal to the amount tendered less the amount charged. But the model also applies to account-based methods. In these methods, the scrip is usually simply credit-card information, and the broker is a credit card company. The difference is that in account-based methods, a consumer's use of scrip to pay for information is actually the exercise of a pre-approved loan.

In all of these methods, the information vendor and the consumer interface directly in the pay-per-access transaction; i.e. the consumer sends a request for particular information with scrip to pay for it, and the vendor provides the information. In doing this, the vendor ensures that the scrip is authentic by checking that the consumer has an account adequate to back the scrip. This requires either that the consumer have an account with the vendor, or that the vendor get clearance to accept the scrip from a central authority. There are three obvious difficulties with such approaches: either a consumer must have a lot of accounts, one with each vendor; or a central authority must be used for each purchase, slowing the access and raising the transaction cost; or, finally, every vendor must develop the capability of handling scrip, which may slow access, depending on what the vendor must do with the scrip.

What is still needed is a method of charging for each access in a way that is secure but fast, and that keeps overhead low. One way to do this is to involve the vendor as little as possible. The interactions that keep costs high or slow access are those needed to ensure that a consumer's scrip is bona fide.

SUMMARY OF THE INVENTION

The method of the present invention uses a third party, called here a banker, as an agent for collecting a fee from a consumer for browsing information provided over a network by a vendor. Before a consumer can request access to a page of information, the banker, after verifying that the consumer has credits on account to pay for the information, mints a token that the consumer can use to pay for the information. The token can be used only by the particular consumer, for

a particular page of information. The banker keeps from the consumer the network location of the information until the consumer returns the token in exchange for the information.

In one embodiment of this method, the interaction between vendor and consumer is limited to the consumer retrieving the vendor-provided information from the vendor after the banker provides the consumer with the search data required to retrieve the paid-for information. The banker manages all other transactions needed to pay the vendor for the information provided. This method is particularly suitable where the network includes the Internet, and where the banker provides the required search data to the user's Internet browser.

To make the information available for a charge, the vendor creates a top commerce (Web) page having links to Web pages the vendor wants to sell, and to any lower-level commerce pages linked to the top commerce page. Each link uses an address to point to a lower-level Web page. Next, for each link on each commerce page, the vendor adds a field that indicates the charge for accessing the linked Web page. Each of these links, now bearing a price for the linked Web page, is called here a priced link.

To introduce the consumer to the top commerce page, the vendor also creates a front door. The front door is a Web page that has an entry link, i.e. a link that ultimately accesses for the consumer a personalized copy of the vendor's top commerce page, giving the consumer access to the pay-per-access Web pages linked directly to the top commerce page, or linked indirectly through lower-level commerce pages that are themselves linked to the top commerce page.

By exercising an entry link, such as by clicking on it, a consumer is connected to the banker, but kept from learning the secret the address of the top commerce page (at the vendor Web site). Exercising the entry link prompts the banker to create a personalized copy of the commerce page, and to provide it to the consumer. Before providing the personalized copy of the commerce page to the consumer, however, the banker alters each priced link by redirecting it to point to the banker, instead of to the vendor, and by attaching a token to it. The priced link is then said to be tokenized. Each token conveys all the information the banker needs to both provide the consumer with, and charge the consumer for, the pay-per-access information associated with the tokenized link.

The method of the present invention is not confined to a particular protocol; for example, it is not limited to a particular set of modifying fields, nor is it limited to a particular manner of embedding fields in an Internet address. In addition, the method of the present invention is not confined to use on the Internet. The essential concept, that of a third-party for minting secure tokens for particular pay-per-access transactions, has general utility, and can be implemented in many ways and for many different applications, as will become apparent to one skilled in the art.

It is an object of the present invention to provide a means of pay-per-access that requires no additional software or hardware for either vendors or consumers.

It is a further object of the present invention to provide pay-per-access in a way that can be implemented without using hypertext commands, electronic wallets, bank cards, or other plug-ins or hardware.

It is further an object of the present invention to provide tokens that are minted only as needed and are good only for a specific transaction in a specific time frame.

It is another object of the present invention to provide a method that permits any facility to become a banker, i.e. to create a market without artificial barriers to entry, so that the number of bankers and the capacity of existing bankers can expand and shrink to accommodate market demand, and each banker can compete with the others for consumer accounts by offering different services in addition to the primary service of managing pay-per-access transactions.

It is also an object of the present invention to keep overhead in charging for access low enough that charges as small as 1/10th of a cent are feasible.

It is another object of the present invention to provide a pay-per-access method in which the consumer's identity can be traced only through the banker.

It is another object of the present invention to provide active measures that prevent a vendor from tricking a consumer into paying more for a link than the consumer believes he is paying.

It is a further object of the present invention to provide pay-per-access requiring no public key technology, or other patented or unexportable cryptography.

It is another object of the present invention to provide pay-per-access enabling automatic content screening so that a consumer can block access to inappropriate information from the consumer's network location, by, for example, the consumer's children.

BRIEF DESCRIPTION OF THE DRAWINGS

The above and other objects, features and advantages of the invention will become apparent from a consideration of the subsequent detailed description presented in connection with accompanying drawings, in which:

FIG. **1***a* is a transaction diagram for pay-per-access methods according to the prior art;

FIG. 1*b* is a transaction diagram for a pay-per-access method according to the present invention;

FIG. **2** is a process flow diagram for a pay-per-access method according to the present invention;

FIG. **3** is a representation of linked pages on the World Wide Web, showing ordinary links and tokenized links;

FIG. 4*a* and FIG. 4*b* illustrate an aspect of the method of the present invention in a case where a consumer and a vendor have accounts with different bankers; and

FIG. **5** is an illustration of an aspect of the method of the present invention in a case where a vendor collects tokens directly from a user.

BEST MODE FOR CARRYING OUT THE INVENTION

The preferred embodiment of the present invention is as a method for paying for information at a Web site of the World Wide Web (WWW), accessed over the Internet.

Some Terminology

In this description, a document is a page of information that a vendor charges to access; a document is located at a Web site, usually that of the vendor charging for access to the document. A page at a Web site can bear information for which a charge is made, or can serve other purposes, such as presenting to a consumer a menu from which to select a document. A consumer is one who views a document on the World Wide Web, called the Web from here on, by accessing the document over the Internet. A link is a part of a document that a consumer can click on, i.e. exercise, to access a related document. A tokenized link is a link that when clicked on causes not only the linked document to be displayed to the consumer, but also results in the consumer being charged a

fee for the document, and the vendor being paid a fee. The fee is paid by the consumer in credit units; the vendor is paid with money; here, the term credit units is used in a generic sense to indicate a data object representation of money, i.e. a kind of scrip. However, a consumer uses a token to convey credit units.

Tokens, which are data objects having a specific value in credit units, are minted by a banker for use in purchasing access to a particular Web page, charging an account of a particular consumer and crediting an account of a particular vendor, usually the vendor that owns the Web page. A banker is any Web-based authority that mints tokens using a protocol based on the present invention and exchanges spent tokens for credit units, which can then be exchanged for ordinary money. A commerce page is a page that includes one or more tokenized links. A tokenized link is a link with an embedded token; exercising a tokenized link, by for example clicking on it, uses the token, which results in access to the particular information for which the token was minted. There is no charge for accessing a top commerce page, i.e. the commerce page uppermost in the vendor's linked hierarchy of pages grouped for a pay-per-access service.

In addition to the terminology specific to the present invention, this description uses some standard terminology associated with Web documents and transport protocols. Hypertext markup language (html) is a format used by most documents on the World Wide Web. A uniform resource locator (URL) is a Web address, which uniquely identifies the location of a particular document (information-bearing page) on the network. A hypertext transfer protocol (http) is the standard protocol used to retrieve and view Web documents. A common gateway interface (cgi) is a Web object that when referenced executes a program at a destination address, and returns some data or a program-generated html document. An extended URL (also called a cgi-extended URL) is a Web address that is said to be extended because of having data used as input by a cgi program at a destination address. An example of an extended URL is

http://www.site.com?data=3991772391.

Setting Up Accounts

Referring now to FIG. 1*b* and FIG. 2, a consumer **14** establishes an account with a banker **12**, receiving a consumer password so that the banker can positively identify the consumer (step **22***a* of FIG. 2). To open the account, the consumer visits the banker's Website, and uses a credit card, for example, to purchase credit units according to some exchange rate geared to allow one credit unit to be worth a fraction of a monetary unit, e.g. a fraction of a U.S. penny (step **22***b* of FIG. 2). The purchased credit units are then kept by the banker in an account for the consumer. In another embodiment, the consumer could be given credit, instead of being required to purchase a specific number of credit units; then the consumer could be billed later for credit units expended.

An information vendor **13** also opens an account with the banker **12** by visiting the banker Website (step **23***a* of FIG. 2). When the vendor opens an account, the banker provides the vendor with an identification the banker can later use to positively identify the vendor. After opening an account, the vendor can make information available to consumers for a charge for each access collected by the banker at the time of access. To allow for this, the vendor specifies the Web address of a Web page he will use as a top commerce page, which includes tokenized links to the pages the vendor wants to sell access to (step **23***b* of FIG. 2). The banker issues the vendor a page-ID for the top commerce page so that the top commerce page can be publicly referenced later without revealing its actual Web address (which is at the vendor's Website).

Embedding Tokenized Links

After establishing an account at a banker, a vendor places in the Web page to be used as a top commerce page one or more links each pointing to a pay-per access page and conveying a cost for access (step **23***c* of FIG. 2). Although other syntax is possible, in the preferred embodiment an ordinary html Web page is made into a commerce page by appending a cost field to a link. For example, if the commerce page has a link,

http://www.vendor.com/news.html

then for it to be parsed as a priced link costing 71 points, the vendor would change the link to

http://www.vendor.com/news.html?cost=71.

In other syntax, the cost field might be added at places other than the end of a link; the only limitation is that the link be parseable.

With the addition of the cost field to a link, the Web page with the link becomes a commerce page, and the link becomes a priced link. Until it is later tokenized by a banker, a priced link points directly to the pay-per-access page at the vendor's Web site. The vendor might also place priced links in lower-level Web pages, making those pages both commerce pages and pay-per-access Web pages. These lower-level commerce pages would then point to information-bearing pages even lower in the linked hierarchy set up by the vendor. If the vendor wants, a lower-level commerce page can serve only as a menu, instead of also bearing pay-per-access information, and the vendor can set the charge to access the lower-level menu-only commerce page to zero (as indeed the charge for any Web page can be set to zero).

Creating a Front Door

The vendor must also create a Web page the consumer can access as a front door to the vendor's store of pay-per-access Web pages (step **23***d* of FIG. 2). The front door leads to a top commerce page, and is needed because the address of the top commerce page must be kept secret from the public, forcing access through a banker. The vendor does this simply by creating a Web page with a link that directs a consumer to a banker, which enables the banker to then prepare a personalized copy of the commerce page for the consumer. For example, to direct a consumer, through a banker, to a vendor's commerce page assigned pageid equal to 51391009 by the banker, the vendor could edit an ordinary link, making it the (cgi-extended URL) entry link:

http://www.banker.com/commerce?pageid=51391009.

Here, commerce is a cgi program, at the banker Web site, that executes when the consumer clicks on the entry link; commerce uses pageid=51391009 as a parameter. The program commerce will tokenize the links of the top commerce page, in the sense explained below, enabling the consumer to purchase information by exercising the tokenized links.

Preparing a Commerce Page for a Consumer

Referring now to FIG. **3** and also to the transaction processing branch of FIG. 2, a pay-per-access begins when a consumer clicks on an entry link of a front door Web page **31**, causing execution of a secure program at a banker **33***a* with a parameter that the banker uses to both look up the address of the commerce page to access, and to identity the

vendor (step **24***a* of FIG. **2**). (The symbol $C_{24a,b}$ of **33***a* of FIG. **3** indicates the banker is performing steps **24***a* and **24***b* of FIG. **2**.) The secure program advises the consumer that the page being accessed includes priced links, and requests the consumer's bank account password. For convenience, the consumer's bank account password could be entered automatically using a browser plug-in program, although this is not necessary.

After the banker **33***a* verifies the consumer's bank account password, it looks up the secret address of the commerce page, and retrieves a copy of the commerce page, either a fresh copy via ftp/http/https, or a cached copy. Next, it tokenizes all priced links on the commerce page, which readies these links for use by the consumer (step **24***b* of FIG. **2**). To tokenize a link, the banker computes a token for the transaction associated with the priced link, and replaces the priced link with a link that instead sends the token to the banker for processing. For example, the priced link,

http://www.vendor.com/news.html?cost=71

would be replaced with

http://www.banker.com/link?token,

where token is a data object banker uses to charge a consumer and credit a vendor account for a pay-per-access. A token is computed as described below in a way that encodes several items of transaction-related information, including at least the amount the consumer should be charged for the information, the consumer identification, the vendor identification, and the address of the destination pay-per-access Web page.

After tokenizing the priced links on the commerce page, the secure program forwards the commerce page to the consumer; this transfer can advantageously be done via https. Throughout all this the network location of the commerce page remains hidden, because the document was forwarded from the banker. Also, at no point thus far has there been a charge to the consumer, because the consumer has yet to exercise a tokenized link; the consumer has so far clicked only on the entry link. Upon viewing the top commerce page, the consumer may decline to purchase any information from the vendor, and there would then be no charge to the consumer.

Making a Sale

With tokens embedded in links on a commerce page, if a consumer clicks on one of the tokenized links, the banker will decipher the embedded token. In one embodiment of the present invention the token may include, among other fields, a signature indicating that the token was in fact issued by the banker, a time-stamp verifying that the token is not too old according to some predetermined allowable time period for using the token, and the number of credit units charged for the information. After deciphering the token, which will provide the banker with all data needed to handle the transaction for which the token was computed, the banker will direct the consumer's browser to the information-bearing, paid-for Web page at the vendor's site. In deciphering a token, the banker examines it for tampering and obsolescence (a token is issued to be good for only a short time), and then extracts the transaction data it needs for charging the consumer account and crediting a vendor account (usually the vendor who owns the information-bearing Web page being accessed).

In another embodiment, the banker protects the consumer by modifying the commerce page to indicate precisely which links are tokenized links by adding a price tag image to each hyperlink that is in fact a tokenized link.

In the case of linking from a commerce page **32** (FIG. **3**) to another commerce page **36**, after performing step **24***c* (FIG. **2**) the banker will restart transaction processing according to step **24***b*, retrieving the commerce page, authorizing all the valid tokenized links, and forwarding the page to the consumer. In linking to a pay-per-access page **34** from commerce page **32**, since there are no tokenized links on the pay-per-access page, the banker **33***b* will perform first step **24***c* of the transaction processing branch, i.e. it will transfer the cost from the consumer to the vendor, and then step **24***b*, directing the consumer's browser to the Web address associated with the tokenized link.

More Commerce Page Syntax

By default, revenue for all tokenized links goes to the vendor identified by the pageid for the top commerce page. However, a vendor could indicate that payment be directed to another vendor by simply adding a vendor field, as for example in:

http://www.vendor.com/news.html?cost=71&vendor=6610012.

Here the standard syntax is used for extending URLs. The question mark character "?" precedes the data, and data items are separated by the ampersand character "&". Other fields used in the preferred embodiment are shown below in Table 1.

TABLE 1

Fields available to a vendor for use in tailoring a tokenized link

Field	Description
cost	cost of a link in credit units
vendor	identification of vendor to be credited
hop	indicates a link leading to another commerce page, a kind of link requiring, in one embodiment, that the identities of the consumer and vendor be preserved
resell	specifies a percentage of each tokenized link transaction a vendor will pay as a commission to another vendor who refers a consumer to a top commerce page
reseller	specifies the vendor identification of the referring vendor
suitability	indicator of suitability for children
reverse	indicates that the direction of the transaction should be reversed, i.e. that the transfer of credit units at the banker is to be from the vendor to the consumer (used for granting refunds)

Computing Tokens

According to the present invention, a token is a data object used by banker to charge a specific consumer for accessing a specific Web document for a specific charge. In the preferred embodiment, a token includes the data fields shown in table 2, where the period "." indicates concatenation of the strings that are the values of the different fields.

TABLE 2

Data fields included in a token

Field	Description
time	seconds since jan. 1, 1970 (standard UNIX timestamp)
consumer	consumer account number
vendor	an identification of a referring vendor
cost	cost for the access (in terms of credit units)
target	address of document to be accessed (enciphered)

From these data fields, the banker constructs a concatenated transcode field:

transcode=time.consumer.vendor.cost.target

In addition to the data fields, a token includes a control variable globalCV, which is, in the preferred embodiment, a

64-hexadecimal-digit, private cryptovariable provided by the vendor's banker. With the control variable and the data fields, the banker constructs a 64-hexadecimal-digit, public signature of the token, according to:

signature=hash(transcode.globalCV)

where hash ( ) indicates any irreversible encipherment algorithm, i.e., any procedure that computes a digital signature for a document so that tampering can be detected. Finally, the banker computes the token by concatenating the signature with the transcode according to:

token=transcode.signature.

If a vendor places other data fields in a link, such as hop or resell, they are also included in the transcode.

When a Consumer Account Falls Short

In the preferred embodiment, whenever a consumer exercises a particular tokenized link on a commerce page, and then backs up from the linked page to the same commerce page, the banker refreshes the commerce page by again tokenizing the links. That way the consumer can be charged again for re-accessing the same information.

If some of the tokenized links on a commerce page cost more than the consumer has on account, but the consumer can afford some of the other tokenized links, the banker will mint the tokens the consumer can pay for. If, after accessing a pay-per-access page, the consumer backs up to the same commerce page, the banker will again tokenize the links according to the same procedure as before.

When a Consumer and Vendor Use Different Bankers

Referring now to FIGS. 4*a* and 4*b*, when a consumer **44** and vendor **41** have accounts at different bankers, the consumer's banker **43** and the vendor's banker **42** cooperate in enabling the consumer to access the vendor's information. After the consumer clicks on an entry link at the vendor's Web site, the vendor's banker retrieves a copy of the vendor's top commerce page and attempts to tokenize its priced links. When the vendor's banker discovers that the consumer does not have an account with it, it asks the consumer what banker to use. Assuming the consumer has an account with some other banker, the consumer provides the vendor's banker with account information the consumer has with another banker. The vendor's banker then tokenizes the priced links on the top commerce page and forwards the page to the consumer's banker. The consumer's banker then also tokenizes the now already once tokenized links.

In other words, the vendor's banker takes a priced link from the top commerce page, such as

http://www.vendor.com/news.html?cost=71

and replaces it with a token it computes, so that it becomes, for example,

http://www.banker-vendor.com/link-vendor?token-vendor,

where token-vendor includes the un-enciphered cost (in credit units) in its transcode element, so that the consumer's banker can read the cost. The vendor's banker then forwards the commerce page to the consumer's banker, which then replaces each once-tokenized link with a link bearing a token the consumer's banker mints. Such a doubly-tokenized link could look like,

http://www.banker-consumer.com/link-consumer?token-consumer,

in which token-consumer encapsulates the token minted by the vendor's banker.

After tokenizing each link on the commerce page, the consumer's banker forwards it to the consumer. This enables the consumer to exercise a tokenized link, by clicking on it for example, prompting the consumer's banker to process (unwrap) the token it minted, charge the consumer and credit the vendor's account, and then redirect the consumer to the vendor's banker. In redirecting the consumer to the vendor's banker, the consumer's banker conveys to the vendor's banker the token minted by the vendor's banker. The vendor's banker unwraps its token, and, based on the fields the token contains, redirects the consumer's browser to the now paid-for information. At the same time, the vendor's banker will credit the vendor's account for the access. Later, it may withdraw from its account with the consumer's banker the credit units deposited there by the consumer's banker when the consumer's banker charged the consumer for the access.

An Embodiment Where the Vendor Collects Tokens

Referring now to FIG. **5**, in another embodiment of the present invention, a vendor **52** collects tokens from a consumer **53**, instead of a banker collecting the tokens. A banker entity **51**, which might be both the vendor's banker and the consumer's banker acting in cooperation as explained above, only mint the tokens for a commerce page, embed them in the commerce page, and provide the commerce page to the consumer. In this embodiment, the location of the vendor's information is still kept secret from the consumer until the consumer pays for the information using a token.

To prepare the vendor to accept tokens, the banker entity **51** provides the vendor with franking software and a 64-bit vendor cryptovariable, specific to the vendor. When the consumer **53** exercises a tokenized link, the vendor **52** franks the token, i.e. tests to determine if the token is counterfeit. If the token is bona fide, the vendor deciphers the token to determine the location of the information the token buys, redirects the consumer to the information, and stores the token so that it can redeem it later with the banker entity.

In the preferred embodiment of implementing this aspect of the invention, two additional fields are used in computing tokens; they are shown in table 3.

TABLE 3

Data fields included in a token to be collected by a vendor

Field	Description
vendorCV	64 random bits, a cryptovariable used to identify a vendor to a banker
mtarget	target enciphered using vendorCV

In tokenizing a priced link, in place of token computed as described above, a banker would use

mtoken=token.mtarget.msignature

where

msignature=hash(token.vendorCV).

In addition, the banker would change the URL of the link so that the token is delivered to the vendor's franking program, instead of to the banker. In the case of hops, however, the banker-directed format described above is used for tokenizing a priced link, not this vendor-directed format. In this embodiment a vendor can now frank arriving tokens, extract the network location of the information purchased by the token, and redirect the consumer to the information. This franking is accomplished without breaching token integrity, without resort to public-key operations, and without involving a banker.

All of the embodiments disclosed here rely on the core feature of the present invention—using a third-party as a toll-gate to control access by a consumer to a vendor's information. It is to be understood that the above-described arrangements are only illustrative of the application of the principles of the present invention. Numerous modifications and alternative arrangements may be devised by those skilled in the art without departing from the spirit and scope of the present invention, and the appended claims are intended to cover such modifications and arrangements.

What is claimed is:

1. A method of charging a consumer for access, over a network, to at least one page of information provided by a vendor at a network site, the method using a third-party acting as a banker for transferring funds associated with said charging from the consumer to the vendor, the consumer having a quantity of credit units on account with the banker, the vendor having an account with the banker for holding credit units that the banker collects from consumers, the vendor having a page accessible over the network to the public that includes a link to a program at the banker and a parameter for the program, the parameter for identifying another network page of the vendor in a way that enables the banker to access a copy of the other network page, the other network page having links to lower-level pages that the vendor charges consumers to access, the method executed when a consumer exercises the link on the page accessible to the public, the method comprising having the banker obtain a copy of the other network page of the vendor, and alter on the page the links to lower-level pages, the links to the lower-level pages indicating a charge in credit units for access to each linked page, said altering including the steps of:

a) minting a token for each page of linked information, the token encoding all of the data the banker needs to charge the consumer for accessing said linked page of information;

b) redirecting each link to the banker site and embedding in each redirected link the token for the linked information; and

c) also embedding in each redirected link instructions for the banker that if the consumer clicks on one of the links, the banker is to execute a program for charging the consumer account for the access in credit units, crediting the vendor account, and directing the consumer to the linked information without identifying to the consumer the network location of the linked page of information,

whereby the consumer accesses a page of linked information only after paying for the information with a token, and without learning the network location of the linked page of information.

2. A method as claimed in claim 1, further comprising the step of determining which pages of information the consumer has enough credit units on account to pay for.

3. A method as claimed in claim 1, wherein the third-party comprises a banker holding an account of the vendor and a different banker holding an account of the consumer, wherein the vendor's banker has an account with the consumer's banker, wherein the altering, on the other network page of the vendor, the links to lower-level pages is performed first by the vendor's banker, creating once-altered links, and wherein the method further comprises the steps in which the consumer's banker:

a) determines which pages of information the consumer has enough credit units on account to pay for;

b) mints a token for each page of linked information, the token encoding all of the data the consumer's banker needs to charge the consumer for accessing said linked page of information;

c) redirects each link to the consumer's banker site and embeds in each redirected link the token for the linked information; and

d) also embeds in each redirected link instructions for the consumer's banker that if the consumer clicks on one of the links, the consumer's banker is to execute a program for charging the consumer account for the access in credit units, crediting the vendor's banker's account, and directing the consumer to the vendor's banker.

4. A method as claimed in claim 1, wherein, when the consumer exercises a link to a lower-level page requiring a token to access, the vendor's banker collects the token from the consumer, determines if the token is valid, and, if so, directs the consumer to the lower-level page.

5. A method as claimed in claim 1, wherein, when the consumer exercises a link to a lower-level page requiring a token to access, the vendor collects the token from the consumer, determines if the token is valid, and, if so, directs the consumer to the lower-level page.

* * * * *

US006023508A

United States Patent [19]

Bombard et al.

[11] **Patent Number: 6,023,508**

[45] **Date of Patent: Feb. 8, 2000**

[54] **POLYMORPHIC DATA STRUCTURES FOR SECURE OPERATION OF A VIRTUAL CASH SYSTEM**

[75] Inventors: **Thomas J. Bombard; John J. Hassett,** both of Marblehead, Mass.

[73] Assignee: **Currency Scientific, Inc.,** Marlbehead, Mass.

[21] Appl. No.: **08/861,628**

[22] Filed: **May 22, 1997**

[51] **Int. Cl.**[7] **H04L 9/32**

[52] **U.S. Cl.** **380/24**; 380/4; 380/21; 705/41

[58] **Field of Search** 380/3, 4, 25; 705/68, 705/69; 713/200; 235/379; 364/408

[56] **References Cited**

U.S. PATENT DOCUMENTS

4,303,904	12/1981	Chasek	340/23
4,630,201	12/1986	White	364/408
4,689,478	8/1987	Hale et al.	235/380
4,906,828	3/1990	Halpern	235/379
5,175,416	12/1992	Mansvelt et al.	235/379
5,420,405	5/1995	Chasek	235/379
5,835,726	11/1998	Shwed et al.	395/200.59

OTHER PUBLICATIONS

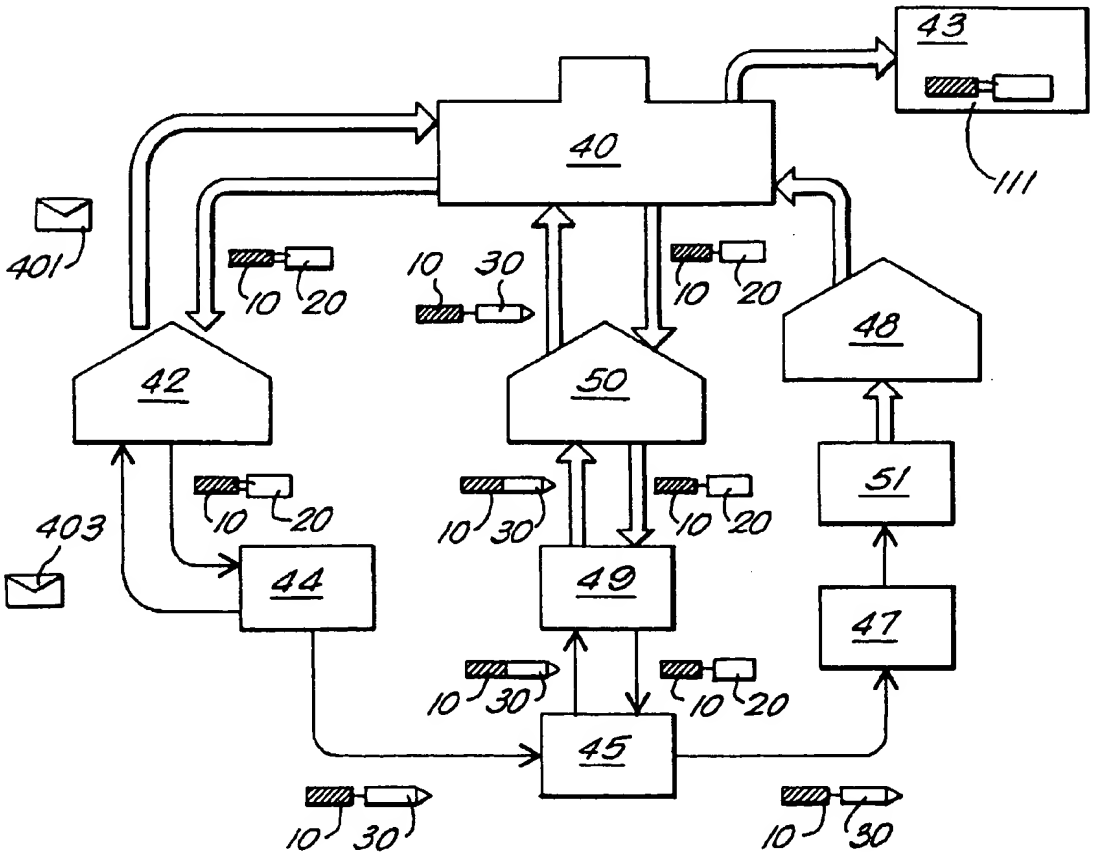
Dickson et al., "European Bank Insight," Smith Barney Report, pp. 2–28 (Apr. 15, 1997).

Primary Examiner—Tod R Swann
Assistant Examiner—Todd Jack
Attorney, Agent, or Firm—Lahive & Cockfield, LLP

[57] **ABSTRACT**

A system for transferring value carrying data packets representative of cash between transferor and transferee terminals without the intervention of a centralized database provides for data packets convertible between an inspected state and an uninspected state. Data packets in an inspected state can be negotiated between terminals once, whereupon they become uninspected and hence, non-negotiable. Uninspected data packets are restored to inspected state by having a central bank compare a hash code generated by a transferor terminal against a corresponding hash code generated by the central bank.

13 Claims, 5 Drawing Sheets



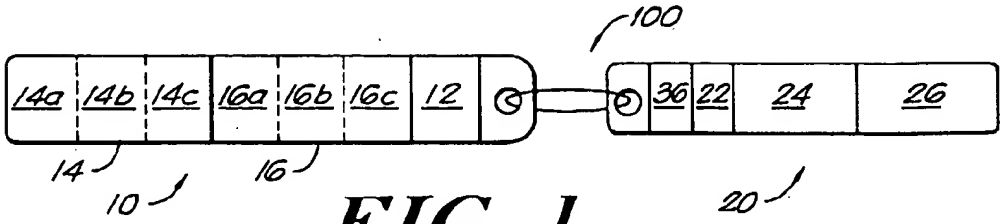


FIG. 1

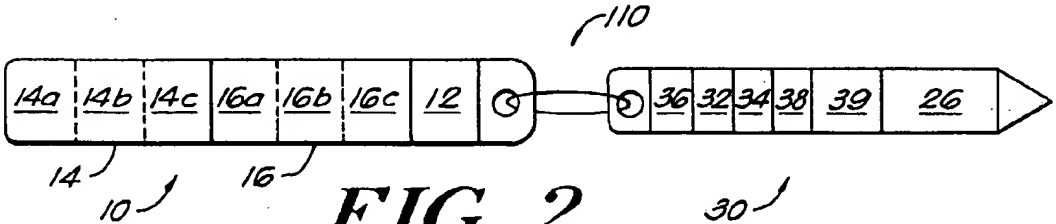


FIG. 2

FIG. 3

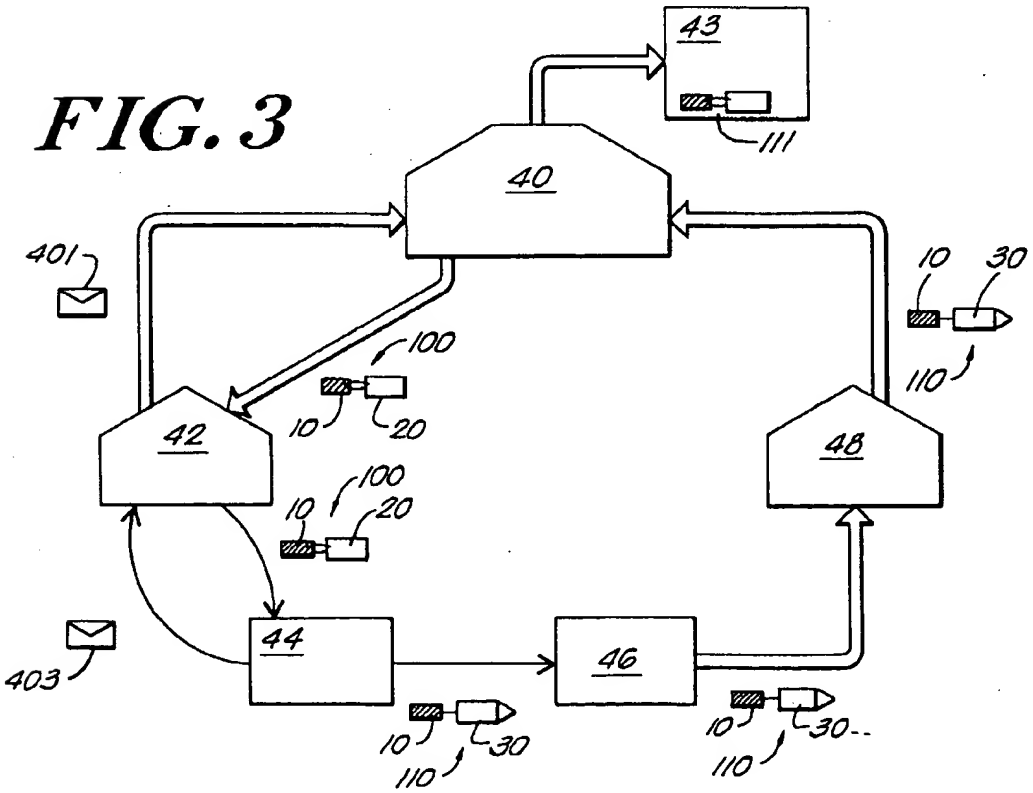
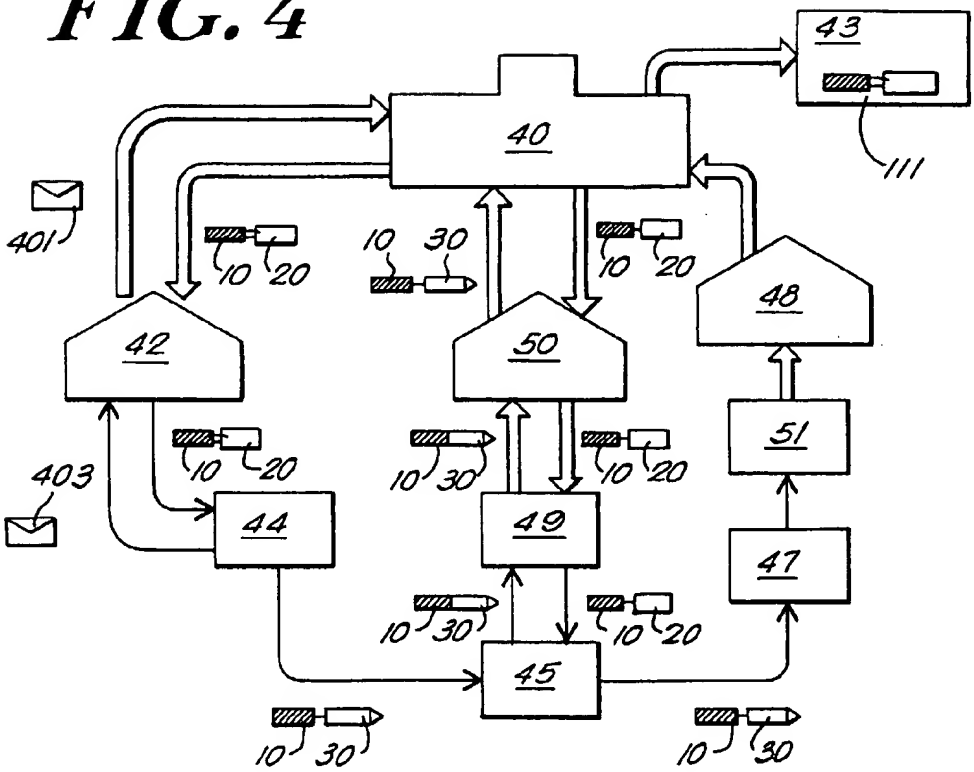


FIG. 4



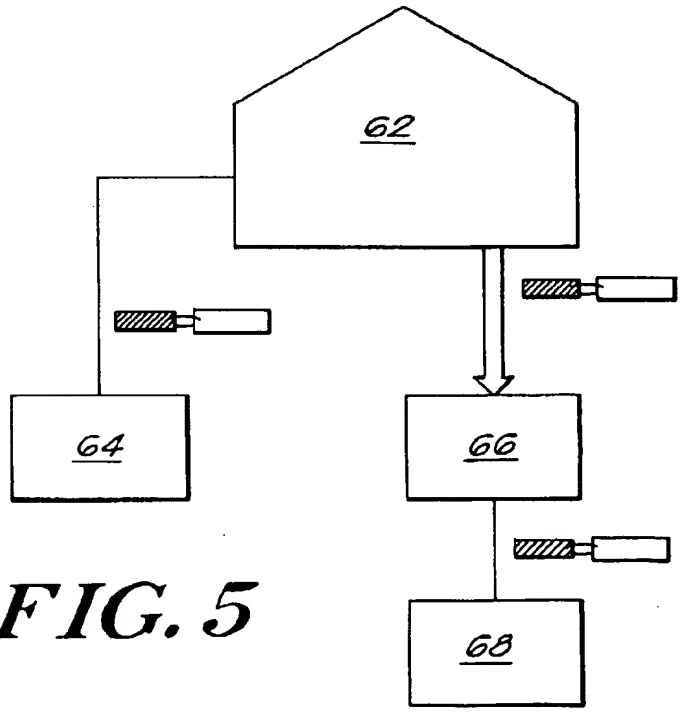


FIG. 5

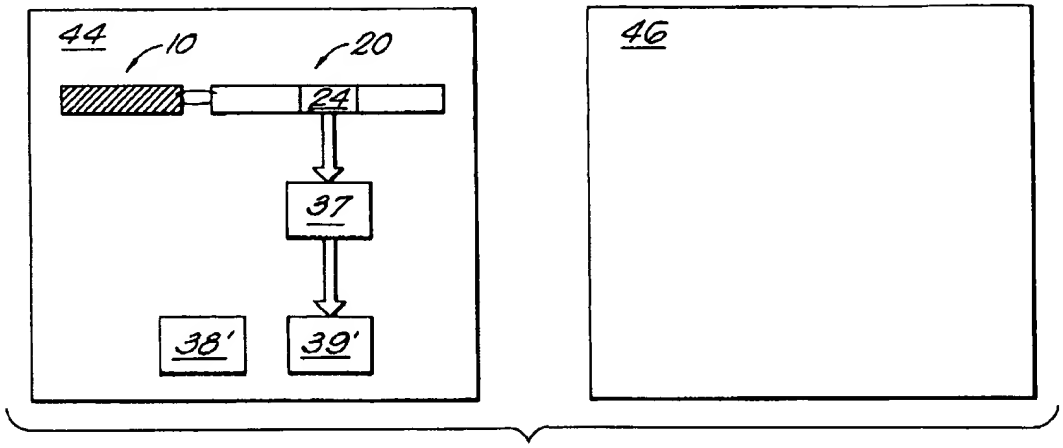


FIG. 6A

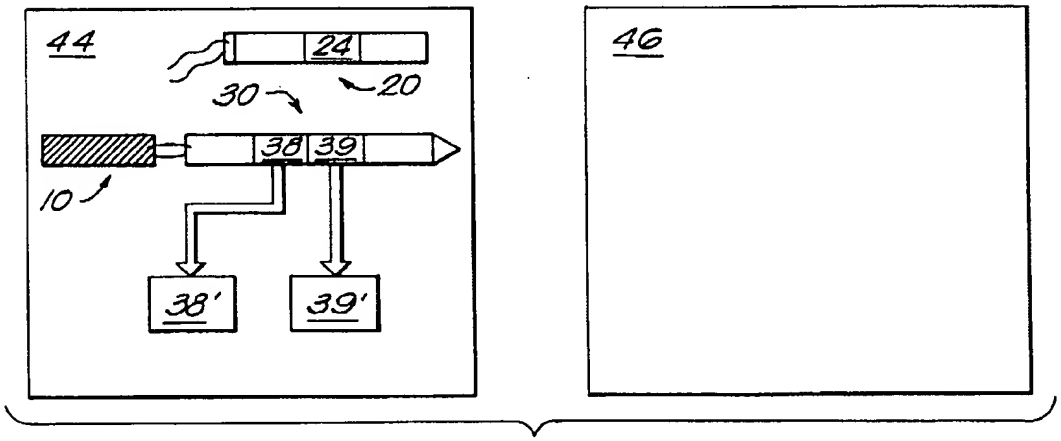


FIG. 6B

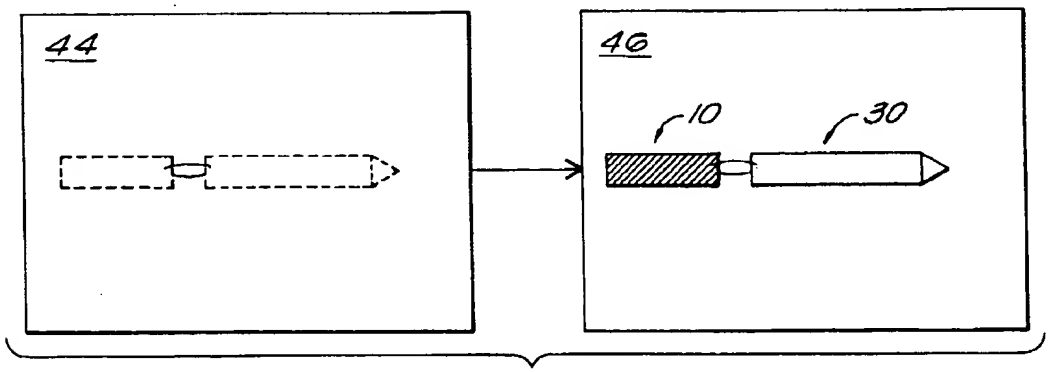


FIG. 6C

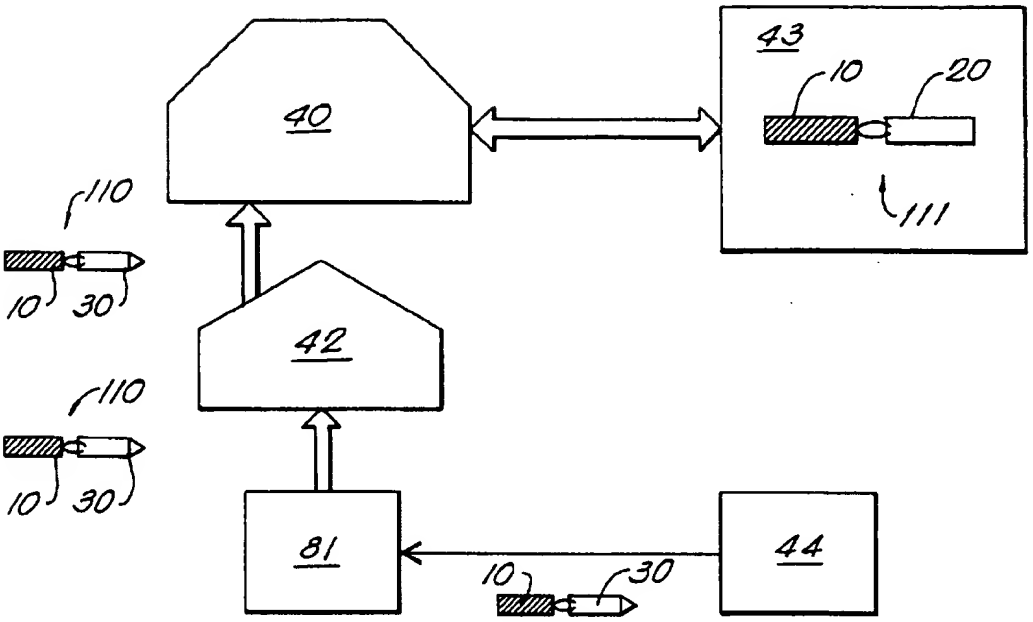

40
43
10
20
111
110
10
30
42
110
10
30
81
44
10
30
110

FIG. 7A

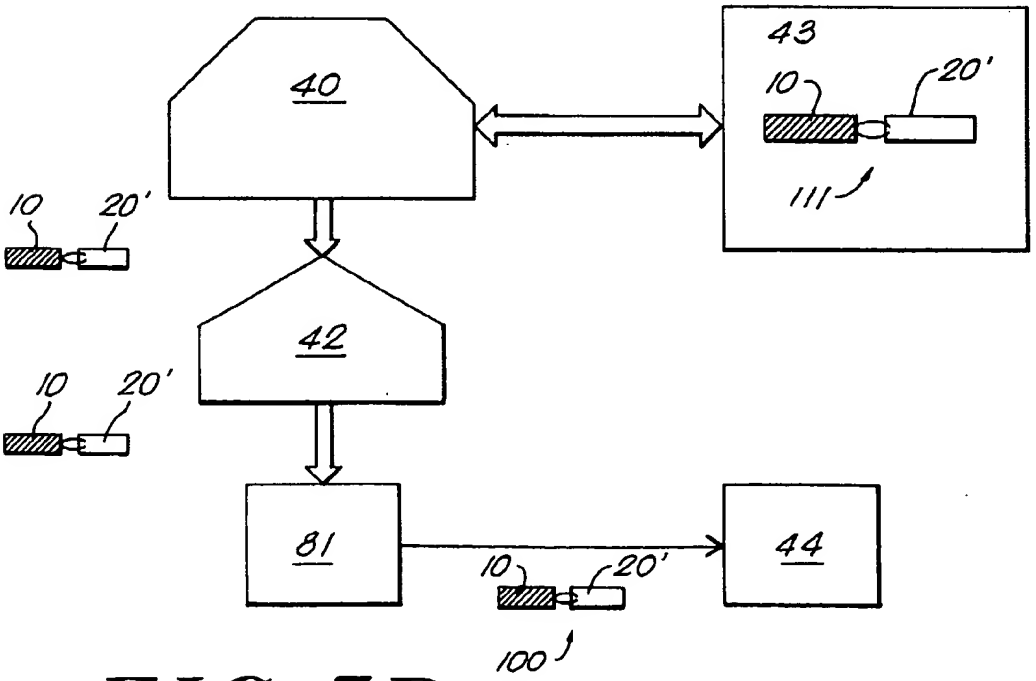

40
43
10
20'
111
10
20'
42
10
20'
81
44
10
20'
100

FIG. 7B

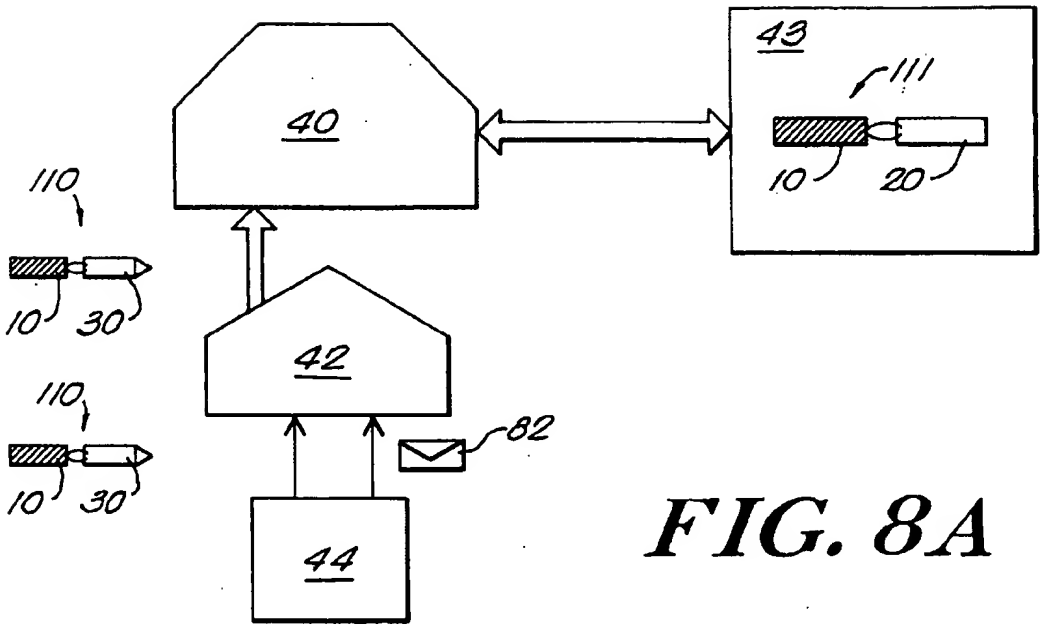


FIG. 8A

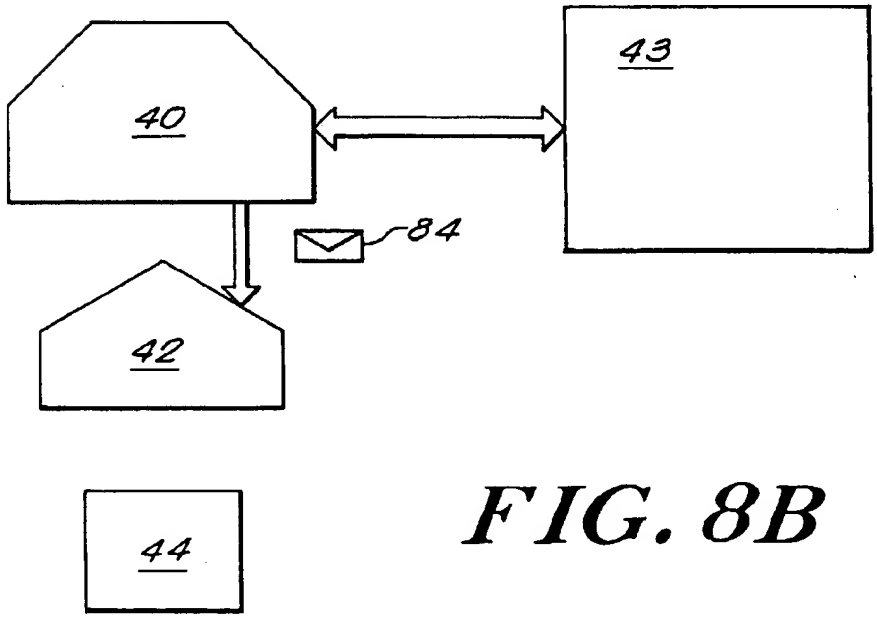


FIG. 8B

POLYMORPHIC DATA STRUCTURES FOR SECURE OPERATION OF A VIRTUAL CASH SYSTEM

This invention relates to a combination of methods which together implement a system for the electronic transfer of value combining the anonymity and decentralized nature of cash transactions with the security and fraud-resistant features of credit card transactions.

BACKGROUND

Systems for transferring value have evolved in two directions. In one direction, represented by credit cards, each value-transferring transaction is individually authorized online by a centralized clearinghouse and recorded at the time of the transaction. Authorization typically consists of verifying whether the account number read from the card has sufficient credit to engage in the proposed transaction. These systems are typically usable for a broad variety of transactions involving different vendors. In the other direction, represented by prepaid phone cards or subway tickets having a remaining value encoded on a magnetic stripe, there is no record of individual transactions. Inspection is performed off-line without a central clearinghouse by verifying that sufficient value for the transaction is physically encoded on the card. These systems typically can be used for transactions involving only a limited number of vendors.

Recently, "smart cards" having a microprocessor embedded in the card have been developed. These permit a variety of accounts to be encoded on a single card. This permits value to be added to as well as removed from a variety of accounts. A smart card is therefore typically equivalent to a plurality of one of the cards described above.

A recent innovation is the Mondex card used to create an electronic form of currency. A system using the Mondex card permits a user of the system to transfer a data packet representative of cash to another user of the system in such a way that the recipient of the data packet can transfer it again to yet another user of the system, for value, or to a bank, for credit to an account. Unlike the credit card systems described above, the Mondex system permits transactions to take place off-line without the intervention of any centralized clearinghouse. Unlike the debit card systems described above, the cash equivalents circulating in a Mondex system can be used to pay a variety of different vendors.

The Mondex system falls short of implementing a true replacement for physical cash. In the Mondex system, cash is never actually minted. It is merely recognized as value being depositable in a bank. In addition, security is limited and rudimentary. There exists no effective way to authenticate the circulating data packets and to detect counterfeits.

There exists a need, therefore, for a system of circulating data packets representative of cash in which an authentication mechanism reliably detects counterfeit data packets.

SUMMARY

The present invention provides for a secure system for circulating a data packet representative of a cash note between a plurality of terminals. The system includes methods for creating and destroying the data packet and methods for negotiating the data packet from a transferor terminal to a transferee terminal in a secure manner. As part of the negotiation process, the system provides a method for periodically inspecting the data packet to assure its authenticity and its integrity.

The data packet representing the cash note can exist in one of two circulation states. In the non-circulating state, transfer of the data packet from one terminal to another is restricted. In the circulating state, the data packet can exist in one of two inspection states: an inspected state, in which it can be freely transferred from one terminal to another, and an uninspected state in which transfer from one terminal to another is restricted.

The data packet includes an encrypted record permanently associated with the note and a cleartext record which can be altered by the terminal having possession of the data packet. It is by suitably altering the cleartext record that a terminal changes the inspection state or the circulation state of the data packet.

The encrypted record includes a serial number, which uniquely identifies the data packet, and an original face value, which indicates the denomination of the cash note. Additionally, the encrypted record can include means to identify the creator of the data packet and means to identify the terminal requesting the data packet. A private key is used to encrypt the encrypted record.

The cleartext record contains a field identifying the inspection state of the data packet, a field specifying the current face value of the data packet, and, optionally, a field specifying the circulation state of the data packet. In the inspected state, the cleartext record includes a key which can be used by a transferor terminal to generate a signature corresponding to that key. In the uninspected state, the cleartext record includes the signature generated by the transferor terminal using the key.

A record of the existence and status of a data packet is stored in a validation database. This validation database is updated during the lifetime of the data packet to reflect changes such as changes in inspection status, circulation status, or in the value remaining in the data packet.

The system includes five types of transactions. These transactions differ in the changes made to either the circulation state or the inspection state of the data packet. The five transactions are:

- minting, which includes generating the data packet's encrypted record and attaching to it a cleartext record which places the data packet into a non-circulating state;
- withdrawing, which includes altering the cleartext record to place the non-circulating data packet into a circulating and inspected state;
- negotiating, which includes altering the cleartext record to place the circulating and inspected data packet into an uninspected state;
- inspecting, which can include, if the data packet passes inspection, altering the cleartext record to place the circulating and uninspected data packet into an inspected state
- depositing, which includes altering the cleartext record to place the circulating data packet into a non-circulating state.

The system includes five types of terminals which differ in the types of transactions they can engage in. These are:

- central banks, which can mint data packets and which maintain validation databases of data packets in circulation;
- account custodians, which can change the circulation state of a data packet and which can restore an uninspected data packet to its inspected state;
- personal terminals, which can change a data packet from being in an inspected state to being in an uninspected state;

home-based terminals, which, by communicating with an account custodian, can restore an uninspected data packet into its inspected state; and

vendor terminals, which can place the data packet into an uninspected state, and which, by communicating with an account custodian, can restore the data packet into its inspected state.

Minting

Minting is the process of creating a data packet representative of a cash note. The process of minting is typically performed by a central bank terminal at the request of another terminal, typically an account custodian. The account custodian usually specifies the cash value of a data packet to be minted. The central bank then checks the account custodian's credit. If the account custodian's credit is deemed satisfactory, the central bank creates the data packet by assembling the parts as described above. The central bank then updates a database to indicate that it has created a new data packet.

The new data packet, in a non-circulating state, is then transferred to the requesting account custodian.

Withdrawing

Withdrawing a data packet is accomplished by having a cash data packet transferred from a dispensing terminal to a withdrawing terminal. Normally, the dispensing terminal is an account custodian or a terminal in communication with an account custodian, such as a home-based terminal.

The dispensing terminal alters the cleartext record of the data packet to be withdrawn in order to place it into a circulating state and into an inspected state. The data packet, now in a circulating and inspected state, is transferred to the withdrawing terminal, typically a personal terminal.

Negotiating

During negotiation of the cash data packet, the transferor terminal, usually a personal terminal, alters the cleartext record of an inspected cash data packet, thereby placing it in an uninspected state, and transmits it to a transferee terminal. The cash data packet will then exist in the transferee terminal, most often another personal terminal, in an uninspected state.

Because a data packet is negotiable only in its inspected state, the transferee terminal will not be able to negotiate the data packet to another terminal until an inspecting terminal inspects the data packet in the manner described below and restores it to its inspected state.

As is often the case in cash transactions, the purchase price of the good may not correspond to the inspected data packets, either singly or in combination, available to pay for the goods. Under these circumstances, it becomes necessary to subdivide the data packets, thereby making available a combination of data packets having a denomination corresponding to the purchase price of the goods. This process, commonly referred to as "making change," is a feature of another embodiment of the invention.

To make change, the data packet representative of a cash data packet is duplicated into a first data packet to be spent and a second data packet to be retained. The value of the data packet to be spent is set to the purchase price of the goods. The value of the data packet to be retained is set to the value of the data packet prior to the transaction less the purchase price. Both the data packet to be spent and the data packet to be retained have the same encrypted record. The data packet to be spent is then negotiated in the manner set forth above.

Inspecting

During inspection of the cash note, the terminal requesting inspection, for example a personal terminal, transfers the note to an inspecting terminal, which could be an account custodian. The inspecting terminal decodes the transferor's signature from the note's cleartext record to recover its corresponding key. This enables the inspecting terminal to recover the note in the form it was in before the transferor terminal transferred it. The inspecting terminal can then compare the recovered note against a copy of the note as stored in a validation database of circulating notes maintained by a central bank terminal. If the stored copy of the note is consistent with that presented for inspection, the inspecting terminal alters the cleartext record of the note to place it back in the inspected state and sends the inspected note back to the terminal requesting the inspection. The inspecting terminal then causes the validation database of circulating notes to be updated with the inspected note so that a copy of the note will be available when that note is next presented for inspection.

Depositing

A depositing terminal, typically a personal terminal or a vendor terminal, can also deposit the note, for credit, into an account at a financial institution. It does so by transferring the note to a terminal in communication with an account custodian, or to the account custodian itself. In one embodiment, the depositing terminal is a personal terminal which transfers the note to a home-based terminal in communication with an account custodian. In another embodiment, the depositing terminal is a vendor terminal which transfers the note directly to an account custodian.

Upon receipt of the note to be deposited, the account custodian causes the note to be removed from circulation, causes the database of circulating notes to be updated to reflect the removal of the note from circulation, and causes the depositing terminal's account to be credited by the amount shown as the current face value of the note being deposited.

These and other features, aspects and advantages of the invention will be better understood with reference to the following description, the appended claims, and the accompanying drawings in which:

FIG. **1** shows a data packet in its inspected state;

FIG. **2** shows the data packet of FIG. **1** in its uninspected state;

FIG. **3** shows the overall circulation of the data packet of FIG. **1** as it makes its way from a minting central bank, to a consumer carrying a personal terminal who then purchases an item from a vendor by using the vendor's vendor terminal;

FIG. **4** shows the overall circulation of the data packet in FIG. **1** with the additional step of negotiation from one personal terminal to another;

FIG. **5** shows two ways in which the data packet of FIG. **1** can enter circulation;

FIGS. **6A–6C** show the steps by which an inspected data packet as shown in FIG. **1** is negotiated from one terminal to another, thereby turning it into the uninspected data packet of FIG. **2**; and

FIGS. **7A–7B** show the steps by which the uninspected data packet of FIG. **6C** can be restored to an inspected state.

DESCRIPTION

A data packet representative of a cash note to be used in a system embodying the invention can exist as an inspected data packet **100**, as shown in FIG. **1**, or as an uninspected data packet **110** as shown in FIG. **2**. In both its inspected state and its uninspected state, the data packet includes two parts: an encrypted cash serial number **10** which remains the same for the life of the data packet; and a cleartext or

unencrypted section whose contents depend on whether the data packet is in an inspected state or an uninspected state. The cleartext portion switches between being a validation check number **20** when the data packet is in an inspected state, as shown in FIG. 1, and a transfer authorization number **30** when the data packet is in an uninspected state, as shown in FIG. **2**.

Referring to FIG. 1, the encrypted cash serial number **10** includes a denomination field **12** indicating the original value of the cash note represented by the data packet and, optionally, the currency used to measure that value. The cash serial number **10** can also include a central bank identification field **14** identifying when and where the data packet was minted. This central bank identification field **14** may include: a central bank identification number **14***a* uniquely identifying the central bank which minted the data packet; a minting time stamp **14***b* showing the date and time that the data packet was minted; and a central bank sequence number **14***c* which can be incremented by the central bank to further identify the data packet.

The encrypted cash serial number **10** can also include an account custodian identification field **16** to identify the account custodian that requested the minting of the data packet. The account custodian identification field **16** can include: an account custodian identification number **16***a* uniquely identifying the account custodian requesting the data packet; a requesting time stamp **16***b* indicating the date and time at which the account custodian requested minting of the data packet; and an account custodian sequence number **16***c* which can be incremented by the account custodian.

In the inspected state, shown in FIG. 1, the unencrypted section of the data packet has a validation check number **20** which includes: an inspection check number **24**, which is an arbitrary number assigned by the central bank upon minting the data packet; a remaining value field **22**, which indicates the value remaining in the data packet and which is always less than or equal to the value shown in the denomination field **12**; and a transfer authorization key **26** which is a random number used to convert a data packet in an inspected state into a data packet in an uninspected state in a manner to be described below.

The validation check number **20** can also include a circulation field **36** for specifying whether the data packet is in a circulating state or in a non-circulating state. In the preferred embodiment, the most significant bit of the remaining value field **22** is used as the circulation field **36** of the data packet.

FIG. **3** shows one possible life cycle of a data packet as it circulates through the stream of commerce. In FIG. **3** an account custodian **42** maintains an inventory of data packets having various denominations for withdrawal by a personal terminal **44**. When the number of data packets having a particular denomination falls below a predetermined threshold, software running on the account custodian for the purpose of monitoring the inventory of data packets triggers a request for additional data packets. The account custodian **42** then transmits the request for data packets to a central bank **40**. In response to this request, software executed by the central bank **40** mints the requested data packets and transmits them to the account custodian **42**. The central bank also executes software instructions to create an entry in a validation database **43** corresponding to the data packet. The account custodian holds each data packet in non-circulating and uninspected form in memory until a personal terminal **44** requests a data packet.

The personal terminal's request for a data packet can originate when a human operator brings the personal terminal **44** into communication with an account custodian **42** and specifies the number and denomination of data packets to the personal terminal through its user-interface. Communication between the personal terminal and the account custodian can be effected by a telephone or by other cable linking them. Communication between the human operator and the personal terminal **44** can occur by providing the personal terminal with a menu system or pointing device, by providing it with a keyboard or keypad which can have predefined keys for common functions, or by leading the human operator through a transaction with on-screen instructions presenting options which can be chosen by means of a keyboard, keypad, or pointing device. In response to the human operator's request, the personal terminal **44** then transmits a message representative of the request to the account custodian **42**.

Upon receiving a request from the personal terminal, software running on the account custodian **42** fetches a data packet from memory and alters the circulation field **36** to indicate that it is now in a circulating state. The account custodian then transmits the data packet to the personal terminal **44**.

The next step occurs when a human operator uses the personal terminal **44** to pay for the purchase of goods or services from a vendor terminal **46**. This can occur when the human operator interacts with the personal terminal's human interface to specify the amount of value to be negotiated to the vendor terminal **46**. The vendor terminal **46** has means to determine that the value negotiated by the personal terminal **44** is consistent with the value of the goods or services being purchased. For example, the human operator of the personal terminal **44** can scan goods across a bar code reader in communication with the vendor terminal **46** thereby communicating the value of the goods to the vendor terminal **46**. The vendor terminal can also accomplish this by having a human operator communicate appropriate instructions through the vendor terminal's user-interface in the same manner that a human operator communicates instructions to a personal terminal **44**. In either case, the vendor terminal **46** execates software instructions required to accept delivery of the uninspected packet from the personal terminal **44**.

The vendor terminal can then deposit the data packet, now in uninspected form as a result of negotiation, by transmitting it to a depository account custodian **48**, which can be either a different account custodian **48**, as illustrated, or the same account custodian **42**. This requires that the vendor terminal execute software instructions to establish communication with the account custodian, to specify the data packets to be deposited, and to specify an account into which value corresponding to those data packets should be credited. This step can be initiated manually, by having a human operator communicate the appropriate instructions through the vendor terminal's user-interface, or automatically at predetermined intervals.

The depository account custodian **48** then transmits the data packet to the central bank **40** for inspection, in a manner set forth below. If the data packet passes inspection, the central bank **40** withdraws it from circulation by deleting its corresponding entry in the validation database and issues a credit to the account specified by the vendor terminal **46**.

FIG. **4** shows another possible life cycle of a data packet. As was the case in FIG. **3**, the data packet enters circulation upon withdrawal from the account custodian **42** by the

personal terminal **44**. The personal terminal **42** then negotiates the data packet to a second personal terminal **45**, thereby placing the data packet in an uninspected state. Before the second personal terminal **45** can negotiate the data packet to a third personal terminal **47**, it must have the data packet inspected. It does so by presenting the data packet to a home based terminal **49** which is in communication with the central bank **40** through an account custodian **50**. The data packet is transferred to the central bank, inspected, and transmitted back to the second personal terminal **45** in inspected form. The second personal terminal can then negotiate the inspected data packet to a third personal terminal **47**. This third personal terminal deposits the data packet for credit to an account by presenting it to another home-based terminal **51** which is in communication with the central bank **40** through a depository account custodian **48** as described in connection with FIG. **3**.

FIGS. **3** and **4** illustrate just two of the many possible life-cycles of a value carrying data packet in the system of the invention. Each life-cycle includes the fundamental steps of minting, withdrawing, negotiating, inspecting and depositing. These fundamental steps and their interactions will be discussed in more detail below.

Upon receiving a minting request **401** from an account custodian **42**, a central bank **40** executes software instructions which verify that the account custodian **42** is sufficiently creditworthy to receive a data packet having the requested denomination. If the central bank deems the account custodian creditworthy, it initiates the minting process by executing additional software instructions which assemble a data packet **100** having an encrypted cash serial number **10** and a validation check number **20** as shown in FIG. **1**.

Referring to FIG. **1**, the validation check number **20** incorporates the information transmitted by the account custodian **42** as part of the minting request **401**. The software instructions executed by the central bank **40** for assembling a data packet initially set the remaining value field **22** to the same value as that stored in the denomination field **12** and set the circulation field **36**, in this case the most significant bit in the remaining value field **22**, to indicate that the data packet is in the non-circulating state.

After assembly, the central bank **40** transmits the data packet **100** to the account custodian **42** and updates a validation database **43** to indicate the existence and current state of the newly minted data packet **100**. The data packet **100** continues to reside, in an inspected and non-circulating state, with the account custodian **42** until the account custodian receives a withdrawal request **403** from a personal terminal **44** specifying a denomination to be withdrawn and an account from which to withdraw it.

When the account custodian **42** receives a withdrawal request **403** from a personal terminal **44**, the account custodian first determines whether the personal terminal is authorized to withdraw a data packet having the requested denomination. This requires that the account custodian determine that the account from which value is to be debited has sufficient value to debit. If the account custodian determines that the personal terminal **44** is authorized to withdraw a data packet having the specified denomination, it chooses a data packet **100** having the requested denomination and changes its circulation status field **36** to indicate that the data packet is in a circulating state. The account custodian **42** then communicates the identity of the data packet placed into circulation to the central bank **40**, thereby triggering the execution of software instructions by the central bank which update the validation database **43**. Finally, account custodian transmits the data packet **100** to the personal terminal **44**.

FIG. **5** depicts two withdrawal mechanisms in the illustrated embodiment. Withdrawal can occur when an account custodian **62** transfers a data packet directly to a personal terminal **64**. Alternatively, withdrawal can occur when a home-based terminal **66** in communication with an account custodian and having a user-interface similar to that of an automated teller machine acts as an intermediary between the account custodian **62** and a personal terminal **68**.

The data packet **100**, now in its circulating and inspected state, resides in the personal terminal **44** until it is "negotiated" to another terminal. Negotiation is the transfer of a data packet from a transferor terminal, typically a personal terminal **44**, to a transferee terminal, typically a vendor terminal **46** as shown in FIG. **3** or another personal terminal **45** as shown in FIG. **4**. A salient feature of the negotiation method of the invention is that it involves only a transferor terminal and a transferee terminal with no intervention by a centralized database. Another salient feature of the negotiation method is that the data packet is transformed from an inspected state **100**, in which it is freely negotiable, to an uninspected state **110**, in which it is non-negotiable.

The negotiation method begins when the transferor **44** uses a cryptographic algorithm **37** and the transfer authorization key **24** of the inspected data packet **100** to generate a hash code as shown in FIG. **6A**. The transferor then replaces the entire validation check number **20** of the inspected data packet **100** with a transfer authorization number **30** as shown in FIG. **6B**. The hash code **39'** generated by the transferor **44** is stored in the transfer hash field **39** of the transfer authorization number **30**. A value to be negotiated **38'** is then stored in the negotiated value field **38** of the transfer authorization number **30**.

Referring to FIG. **2**, the transfer authorization number **30** further includes: the personal terminal sequence number **32**; a personal terminal time stamp **34** indicating the date and time of negotiation; and an inspection check number **26** which is an arbitrary number assigned by the central bank upon minting the data packet.

In many cases, the value to be negotiated **38'** is less than the value found in the remaining value field **22**. When this occurs, the transferor terminal replaces the original data packet with a data packet to be negotiated and a data packet to be retained, each having the same encrypted cash serial number **10**. The transferor terminal then decrements the remaining value field **22** of the data packet to be retained by the value to be negotiated **38'** and sets the negotiated value field **38** of the data packet to be negotaiated to the value to be negotiated **38'**. The data packet to be negotiated is then treated as set forth above.

The negotiated data packet, now in its uninspected state, is then transferred to the transferee terminal **46** as shown in FIG. **6C**. The transferee terminal **46** shown in FIG. **6C** can be a vendor terminal **46**, as shown in FIG. **3**, or another personal terminal **45** as shown in FIG. **4**. In either case, the transferee terminal **46** now has two choices: it can deposit the data packet into an account at a financial institution, as illustrated by FIG. **3**, or it can further negotiate the data packet, as illustrated by FIG. **4**.

Referring to FIG. **4**, if a transferee terminal, in this case a second personal terminal **45**, chooses to further negotiate the data packet, it must first have the data packet "inspected." Inspection is the process of verifying: that an entry corresponding to the data packet to be inspected exists

in the validation database **43**; that the transfer hash generated by the transferor terminal **44** and stored in the transfer hash field **39** is consistent with both the transfer authorization key **24** found in the validation database **43** and with the value **38'** placed by the transferor terminal in the negotiated value field **38**; and that the value transferred is not in excess of the value stored in the data packet's denomination field **12**.

FIG. **7A** shows a typical configuration of terminals for inspecting a data packet. A personal terminal **44** is connected to an inspection terminal **81**. An inspection terminal can be an account custodian, a home based terminal linked to an account custodian, or a vendor terminal linked to an account custodian. The inspection terminal **81** typically is linked to a central bank **40** by a secure communication channel.

Referring to FIG. **7A**, the transferee terminal **44** initiates the inspection process by presenting the uninspected data packet **110** to an inspecting terminal **81**. The inspecting terminal then presents the data packet **110** to the central bank **40** by way of an account custodian **42**. The central bank **40** locates an entry **111** corresponding to the data packet in the validation database **43**. This entry **111** will be a copy of the data packet as it appeared when it was last in an inspected state. As such, it will have the same cash serial number **10** as the uninspected data packet **110**. However, since the entry **111** is a copy of the data packet in its inspected state, it will have a validation check number **20** in place of the uninspected data packet's transfer authorization number **30**.

If the central bank **40** does not find a corresponding entry **111** in the validation data base **43**, it rejects the data packet **110** by sending it back to the transferee terminal **44** in an uninspected state.

If the central bank does find a corresponding entry **111**, it verifies that the negotiated value **38** in the data packet's transfer authorization number **30** is less than or equal to the original denomination **12** of the data packet.

Upon passing the above two tests, the central bank then verifies that the contents of the transfer hash field **39** contained in the uninspected data packet's transfer authorization number **30** correspond both to the transfer authorization key **24** stored in the corresponding entry **111** in the validation database **43** and to the negotiated value stored in the negotiated value field **38** of the uninspected data packet. It does so by evaluating the inverse of the cryptographic algorithm **37** used by the transferor terminal and verifying that the hash code in the transfer hash field **39** generates the correct transfer authorization key **24** and a negotiated value consistent with that found in the negotiated value field **38** of the uninspected data packet.

Failure of any one of the foregoing tests by the data packet **110** results in rejection of the data packet by the central bank **40**. The rejected data packet **110** is then returned to the transferee terminal **44** in an uninspected state.

If the central bank **40** is satisfied with the authenticity of the data packet **110** being inspected, it transforms it into an inspected state **100**. To do so, the central bank **40** replaces the data packet's transfer authorization number **30** with a new validation check number **20'** as shown in FIG. **7B**. This validation check number is typically different from the validation check number **20** attached to the data packet when it was last in an inspected state. The central bank then updates the validation database by replacing the data packet's previous validation check number **20** with this new validation check number **20'**. Finally, the central bank **40** transmits the data packet **100**, now restored to an inspected state, back to the transferee terminal **44** by way of the inspecting terminal.

As an alternative to further negotiating the data packet, the transferee terminal can surrender the data packet in exchange for credit in an account at a financial institution equal to the value specified by the negotiated value field **39** in the transfer authorization number **30** of the data packet. This transaction, shown in FIGS. **8A** and **8B**, begins when the transferee terminal presents the data packet **110**, together with instructions **82** on what account to credit, to an account custodian **42**. The account custodian **42** surrenders the data packet to the central bank **40** with instructions to withdraw the data packet from circulation and to credit the account custodian **42** with value corresponding to that encoded in the data packet **110**. The central bank **40** then inspects the data packet **110** as set forth above. If the data packet **110** passes inspection, the central bank **40**: credits the account custodian **42** with the value encoded in the data packet; removes the data packet's corresponding entry **111** from the validation database **43**; and transmits a message **84** to the account custodian **42** to indicate that its account has been credited. The account custodian then credits the account specified by the instructions **82** from the transferee terminal **44**.

Having described the invention, what is claimed as new and secured by Letters Patent is:

1. A data packet created by an issuing module at the request of a requesting module, said data packet being representative of a cash note and comprising

encrypted identifying means for uniquely identifying said data packet, said encrypted identifying means including a representation of an original face value;

cleartext identifying means including a representation of a current face value less than or equal to said original face value and specifying an inspection state of said data packet, said inspection state being switchable between an inspected state having a transfer authorization key and an uninspected state having a hash code derived from said transfer authorization key.

2. A data packet representative of a cash note and adapted for secure transfer from a transferor terminal having a signature to a transferee terminal, said data packet comprising

encrypted identifying means permanently identifying said data packet;

cleartext identifying means specifying an inspection state switchable between an uninspected state in which said transfer from said transferor terminal to said transferee terminal is restricted and an inspected state in which said transfer from said transferor terminal to said transferee terminal is unrestricted.

3. The data packet of claim **2** wherein said cleartext identifying means includes a field in which said transferor terminal can affix said signature.

4. The data packet of claim **3** wherein said signature comprises a hash code derived from said cleartext identifying means in its inspected state.

5. A method for inspecting a data packet in possession of a holder upon transmission of a request from said holder to an inspector having an associated inspection database containing records of valid data packets, said data packet representative of a cash note and having

a first encrypted identification means permanently identifying said data packet, said encrypted identification means including a representation of an original face value;

a first cleartext identification means specifying an inspection state switchable between an inspected state and an uninspected state, said cleartext identification means indicating that said data packet is in an uninspected state;

a corresponding data packet in said inspection database, said corresponding data packet having a second encrypted identification means identical to said first encrypted identification means and having a second cleartext identification means indicating that said corresponding data packet is in an inspected state, said second cleartext identification means having means to independently generate said first cleartext identification means;

said method comprising the steps of:

transferring said data packet from said holder to said inspector,

verifying that said data packet has a corresponding data packet in said inspection database,

verifying that said data packet is in an uninspected state,

verifying that said current face value of said data packet is less than or equal to said original face value of said data packet,

independently generating said first cleartext record from said second cleartext record,

verifying that said first cleartext record and said independently generated first cleartext record are identical,

altering said first cleartext record to indicate that said data packet is in an inspected state, and

replacing said second cleartext record with said altered first cleartext record,

transmitting said data packet from said inspector to said holder,

such that said data packet in possession of said holder includes a cleartext record identifying said data packet as an inspected data packet.

6. The method of claim **5** wherein said step of independently generating said first cleartext record from said second cleartext record comprises the steps of:

reading an identifying field from said first cleartext record,

reading a transfer authorization key from said second cleartext record, and

applying a hash coding function to said identifying field and said transfer authorization key to generate a hash code, said hash code being identical to a corresponding hash code on said first cleartext record.

7. The method of claim **6** wherein said step of altering said first cleartext record to indicate that said data packet is in an inspected state comprises the steps of:

deleting said corresponding hash code from said first cleartext record,

deleting said identifying field from said first cleartext record,

generating a transfer authorization key, and

adding said transfer authorization key to said first cleartext record.

8. A method for negotiating a data packet in the possession of a transferor module having data processing means to a transferee module, said data packet representative of a cash note having

an inspection state switchable between an uninspected state and an inspected state and set to the inspected state;

an encrypted record uniquely identifying said data packet; and

a cleartext record specifying said inspection state of said data packet, said cleartext record including a transfer authorization key;

method comprising the steps of:

reading said transfer authorization key, with said data processing means, from said cleartext record,

writing with said data processing means, an identifying field to said cleartext record,

generating with said data processing means, a signature based on said transfer authorization key and said identifying field,

writing, with said data processing means, said signature to said cleartext record, thereby indicating that said data packet is in an uninspected state,

transmitting said data packet to said transferor module such that said inspected data packet in possession of said transferor module becomes an uninspected data packet in possession of said transferor module.

9. A system for the secure transfer of a data packet representative of a cash note having

an encrypted record uniquely identifying said data packet,

a cleartext record specifying an inspection state switchable between an inspected state and an uninspected state, and a circulation state switchable between a circulating state and a non-circulating state,

said system comprising:

a central bank terminal for issuing said data packet, said central bank terminal maintaining a record for said data packet and having means to authenticate said data packet and means to alter said inspection state of said data packet from being in an uninspected state to being in an inspected state;

an account custodian terminal for receiving said data packet issued by said first database system, said account custodian terminal having means to alter said circulation state of said data packet from being in a circulating state to being in a non-circulating state;

a personal terminal for receiving said data packet from said account custodian system, said personal terminal having means to alter said inspection state of said data packet from being in an inspected state to being in an uninspected state

such that said data packet issued by said central bank terminal is placed into circulation by said account custodian terminal upon being transferred by said account custodian terminal to said personal terminal.

10. A method for negotiating a data packet from a transferor module to a transferee module, said data packet representative of a cash note and having:

an inspection state switchable between an inspected state and an uninspected state and set to an inspected state,

an encrypted record uniquely identifying said data packet, and

a cleartext record specifying said inspection state,

said method comprising the steps of:

generating a signature based on said cleartext record;

changing said inspection state of said data packet by writing said signature to said cleartext record;

transmitting said data packet to said transferee module.

11. A method for circulating a data packet representative of a cash note, said data packet being switchable from an inspected state to an uninspected state, said method comprising the steps of:

transferring said data packet, in an inspected state, from a central bank to an account custodian,

transferring said data packet from said account custodian to a first terminal,

changing said data packet at said first terminal from an inspected state to an uninspected state,

transferring said uninspected data packet to a second terminal,

changing said uninspected data packet at said second terminal into an inspected data packet,

transferring said inspected data packet from said second terminal to said account custodian,

transferring said inspected data packet from said account custodian to said central bank, and

removing said data packet from circulation.

12. A method for inspecting a data packet in possession of a holder, said data packet representative of a cash note and having

an inspection state switchable between an inspected state and an uninspected state and

a signature generated by said holder, said signature being based on said inspected state of said data packet,

said method comprising the step of

verifying that said signature is consistent with said inspected state of said data packet,

altering said data packet to indicate that said data packet is in an inspected state so that said data packet in possession of said holder is in an inspected state.

13. A system for the secure transfer of a data packet representative of a cash note, said data packet having

an encrypted record uniquely identifying said data packet,

a cleartext record specifying an inspection state switchable between an inspected state and an uninspected state,

said system comprising:

a central bank system for issuing said data packet, said central bank system maintaining a record for said data packet and having means to authenticate said data packet and means to alter said inspection state of said data packet from being in an uninspected state to being in an inspected state,

a terminal for receiving said data packet from said central bank system, said terminal having means to alter said inspection state of said data packet from being in an inspected state to being in an uninspected state.

* * * * *

US006223291B1

(12) **United States Patent**
Puhl et al.

(10) **Patent No.: US 6,223,291 B1**
(45) **Date of Patent: Apr. 24, 2001**

(54) **SECURE WIRELESS ELECTRONIC-COMMERCE SYSTEM WITH DIGITAL PRODUCT CERTIFICATES AND DIGITAL LICENSE CERTIFICATES**

(75) Inventors: **Larry C. Puhl**, West Dundee; **Dean H. Vogler**, Algonquin; **Ezzat A. Dabbish**, Cary, all of IL (US)

(73) Assignee: **Motorola, Inc.**, Schaumburg, IL (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

(21) Appl. No.: **09/277,304**

(22) Filed: **Mar. 26, 1999**

(51) **Int. Cl.**[7] **H04L 13/00**; H04L 9/00
(52) **U.S. Cl.** **713/201**; 713/201
(58) **Field of Search** 713/200, 201, 713/202

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,138,712 * 8/1992 Corbin 713/200
5,260,999 * 11/1993 Wyman 380/4
5,579,222 * 11/1996 Bains et al. 395/712
5,671,412 * 9/1997 Christiano 707/104
5,758,069 * 5/1998 Olsen 713/201
5,758,088 * 5/1998 Bezaire et al. 709/232

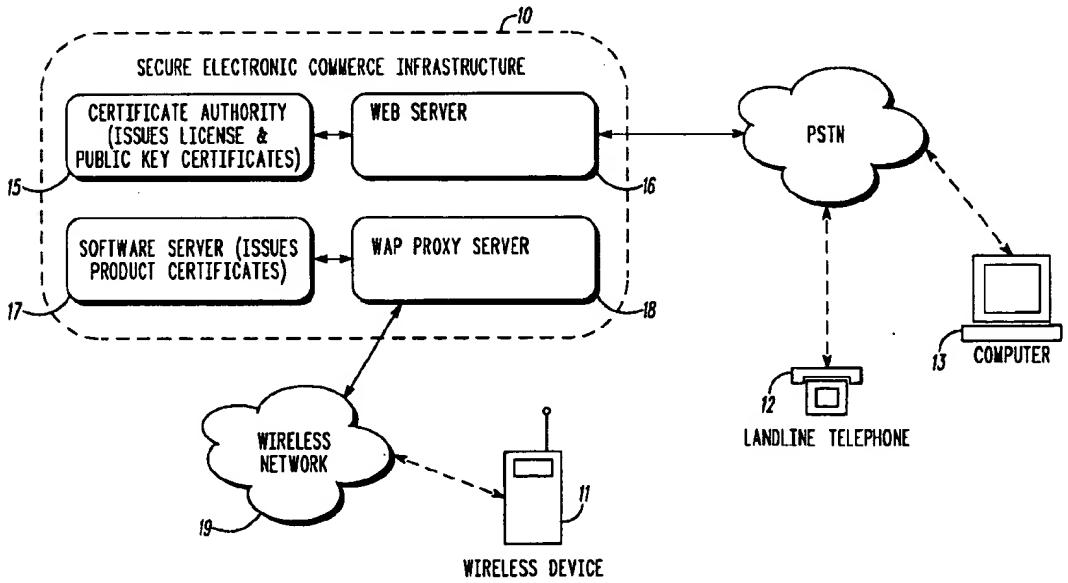
* cited by examiner

Primary Examiner—Ly V. Hua
(74) *Attorney, Agent, or Firm*—Hugh C. Dunlop; Romi N. Bose

(57) **ABSTRACT**

A wireless electronic commerce system (**10**) comprising a wireless gateway (**18**) to a wireless network (**19**) with which a wireless device (**11**) having a unique client identifier (ID) is capable of communicating. A server (**15**) or servers (**15** and **16**) is/are coupleable to the wireless gateway, delivering content items (e.g. software products) to the wireless device (**11**) and maintaining digital content certificates for content items and digital license certificates for licenses for the content items. The server maintains, for each wireless client associated with the system, a record of licenses for that client and a record of content items associated with each license.

7 Claims, 5 Drawing Sheets



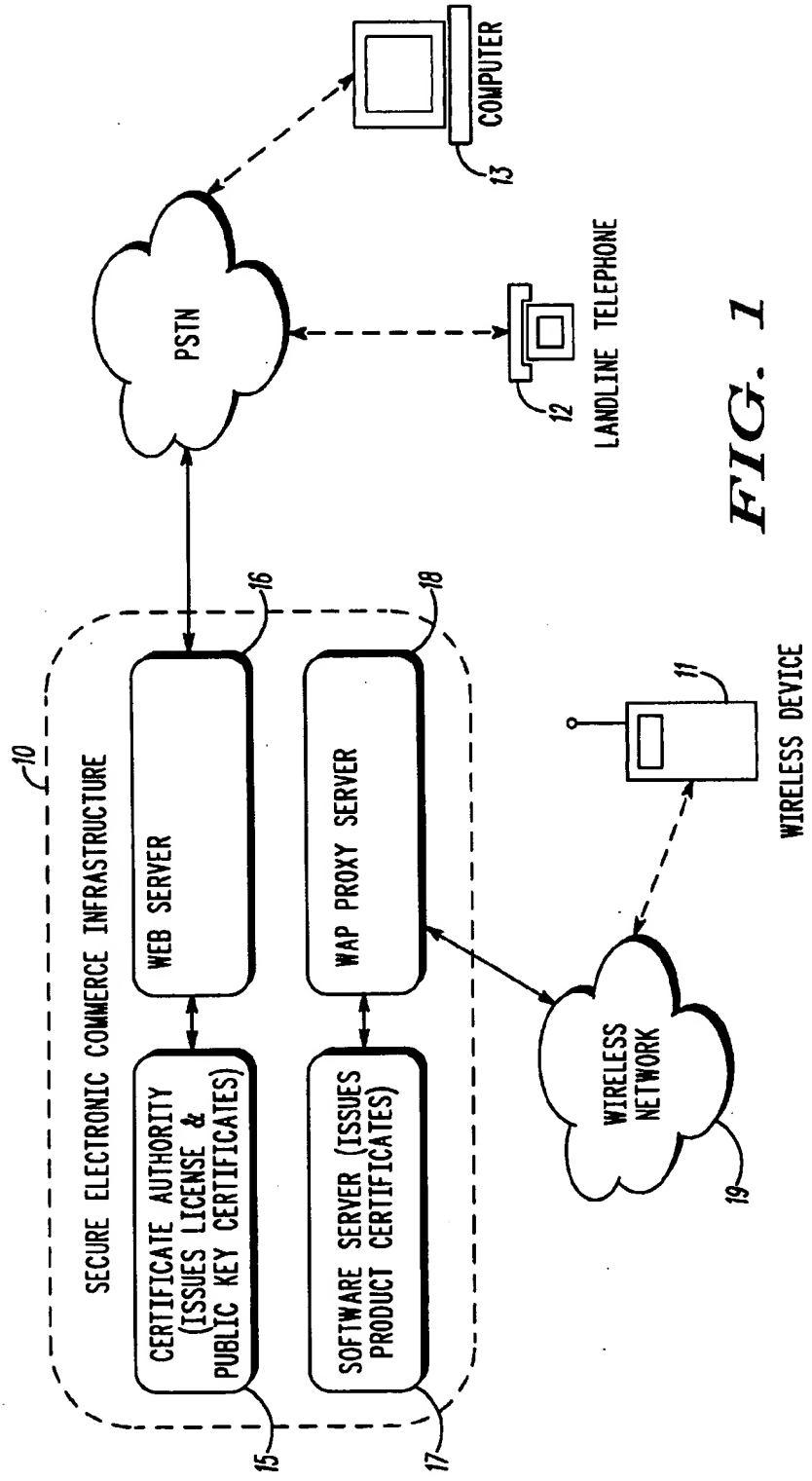

COMPUTER
13
PSTN
LANDLINE TELEPHONE
12
10
SECURE ELECTRONIC COMMERCE INFRASTRUCTURE
WEB SERVER
16
WAP PROXY SERVER
18
CERTIFICATE AUTHORITY (ISSUES LICENSE & PUBLIC KEY CERTIFICATES)
15
SOFTWARE SERVER (ISSUES PRODUCT CERTIFICATES)
17
WIRELESS NETWORK
19
11
WIRELESS DEVICE

FIG. 1

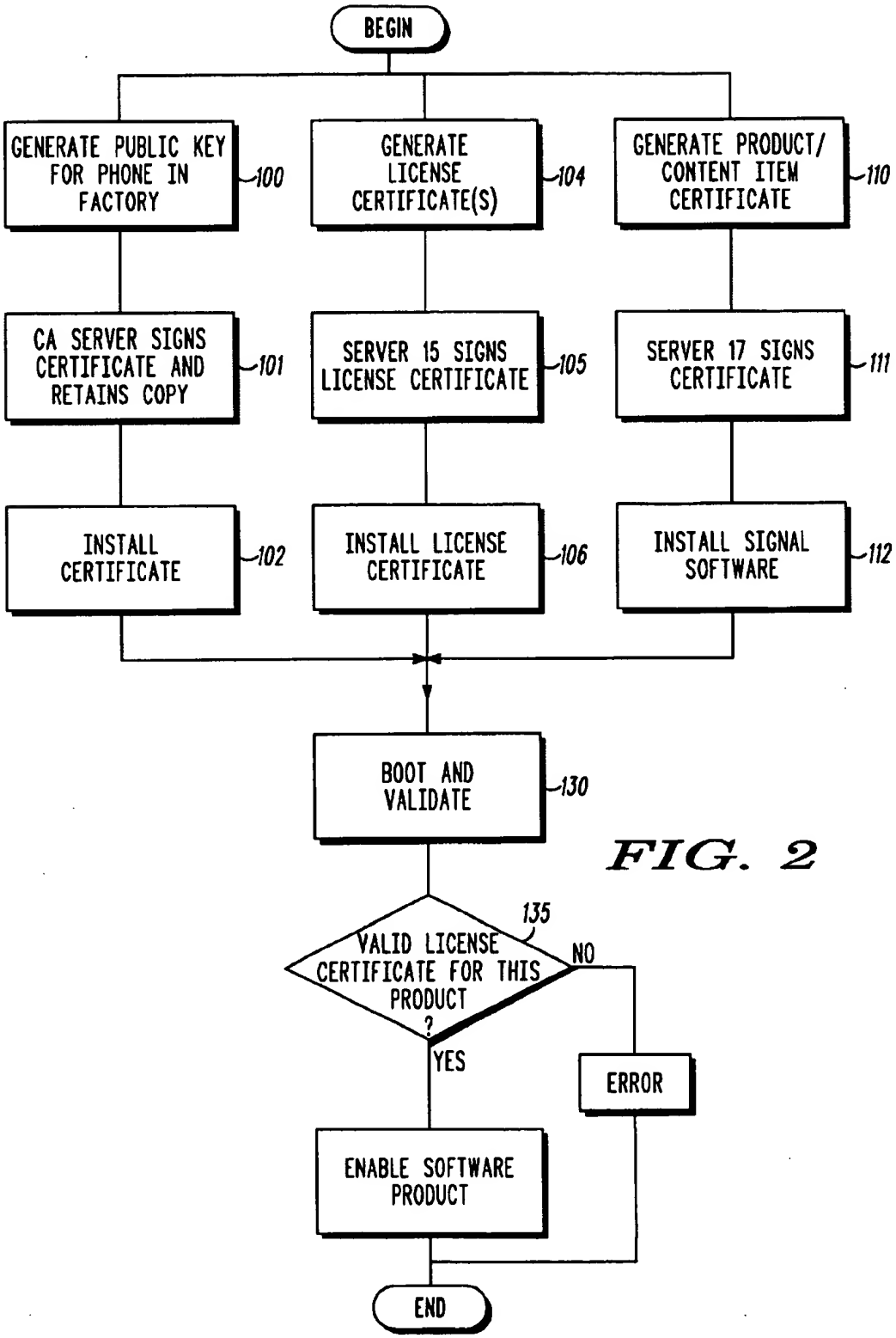

BEGIN
GENERATE PUBLIC KEY FOR PHONE IN FACTORY
100
GENERATE LICENSE CERTIFICATE(S)
104
GENERATE PRODUCT/ CONTENT ITEM CERTIFICATE
110
CA SERVER SIGNS CERTIFICATE AND RETAINS COPY
101
SERVER 15 SIGNS LICENSE CERTIFICATE
105
SERVER 17 SIGNS CERTIFICATE
111
INSTALL CERTIFICATE
102
INSTALL LICENSE CERTIFICATE
106
INSTALL SIGNAL SOFTWARE
112
BOOT AND VALIDATE
130
135
VALID LICENSE CERTIFICATE FOR THIS PRODUCT ?
NO
YES
ERROR
ENABLE SOFTWARE PRODUCT
END

FIG. 2

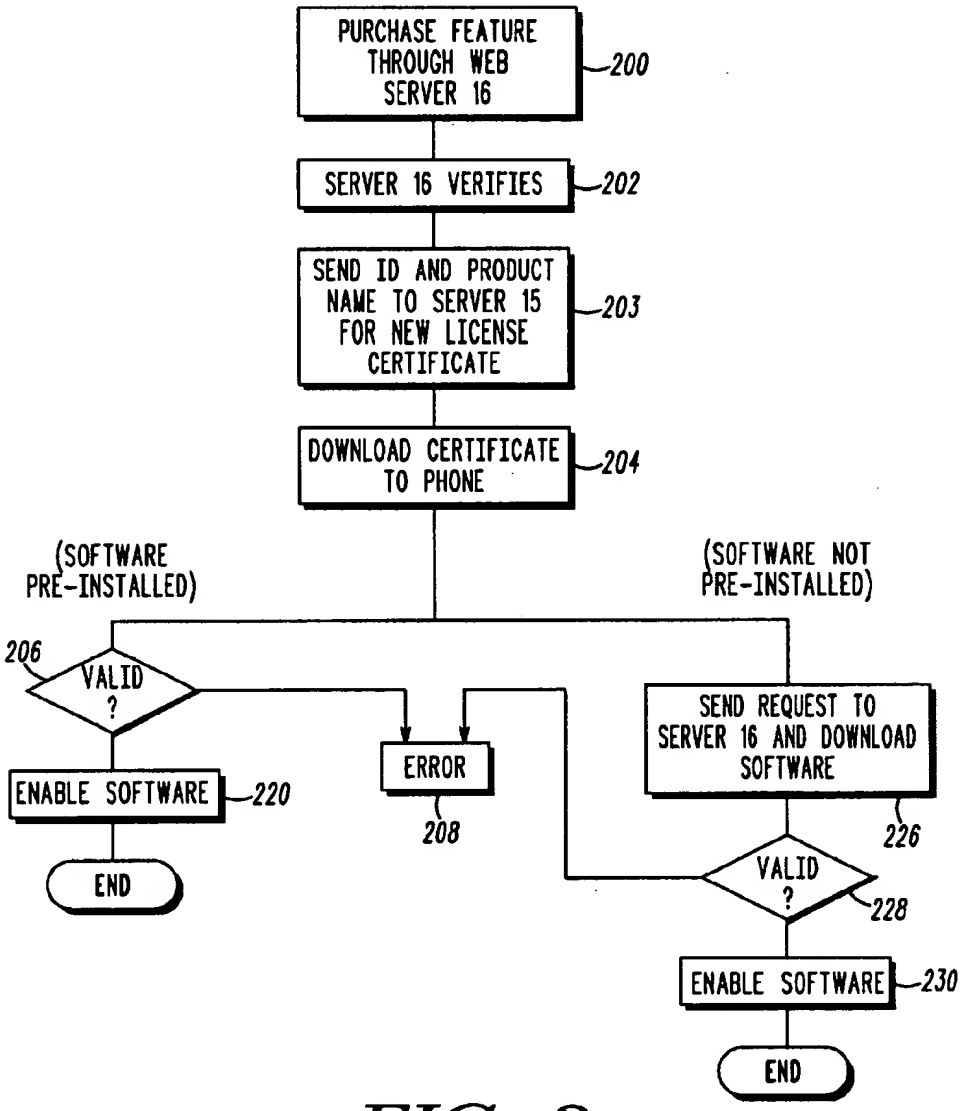

PURCHASE FEATURE THROUGH WEB SERVER 16
200
SERVER 16 VERIFIES
202
SEND ID AND PRODUCT NAME TO SERVER 15 FOR NEW LICENSE CERTIFICATE
203
DOWNLOAD CERTIFICATE TO PHONE
204
(SOFTWARE PRE-INSTALLED)
(SOFTWARE NOT PRE-INSTALLED)
206
VALID ?
ENABLE SOFTWARE
220
END
ERROR
208
SEND REQUEST TO SERVER 16 AND DOWNLOAD SOFTWARE
226
VALID ?
228
ENABLE SOFTWARE
230
END

FIG. 3

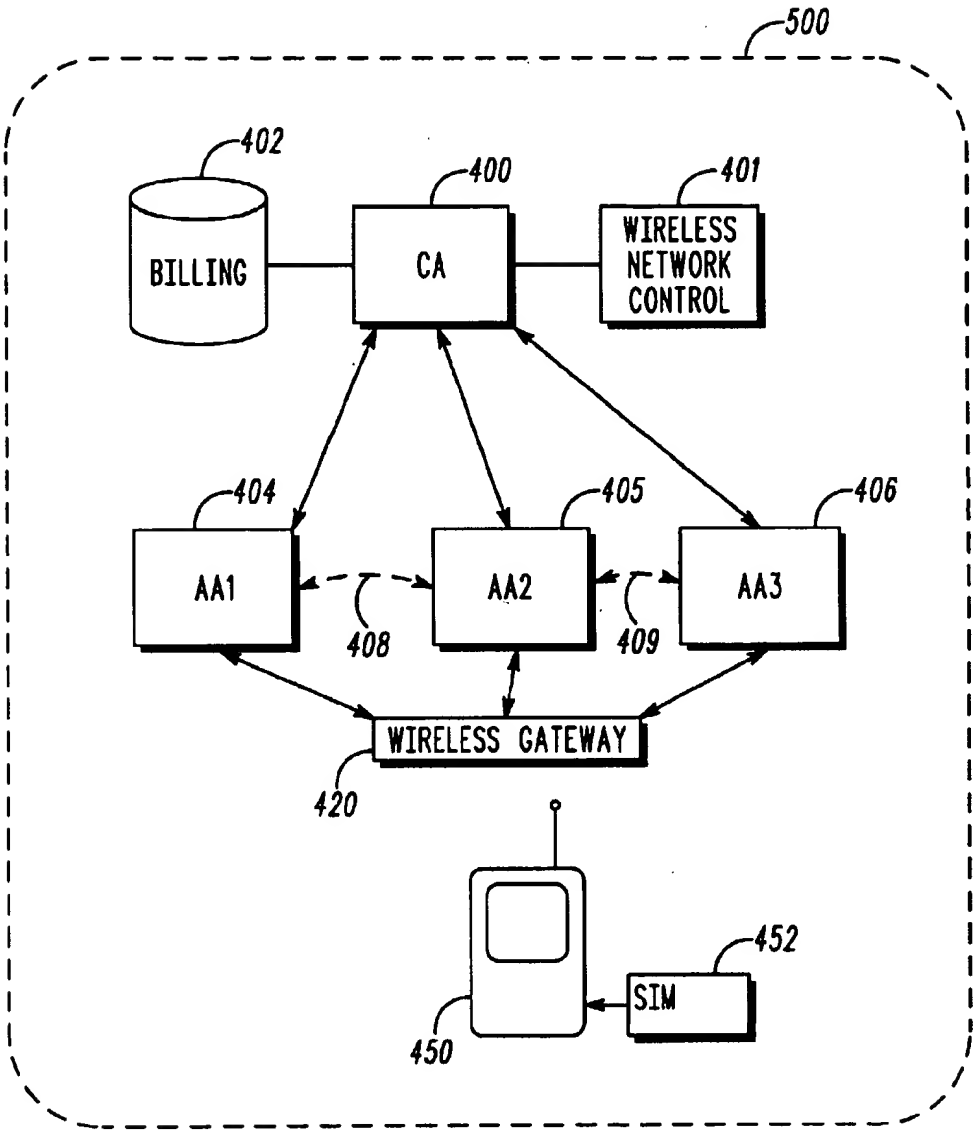

500
402
400
401
BILLING
CA
WIRELESS NETWORK CONTROL
404
405
406
AA1
AA2
AA3
408
409
WIRELESS GATEWAY
420
452
SIM
450

FIG. 4

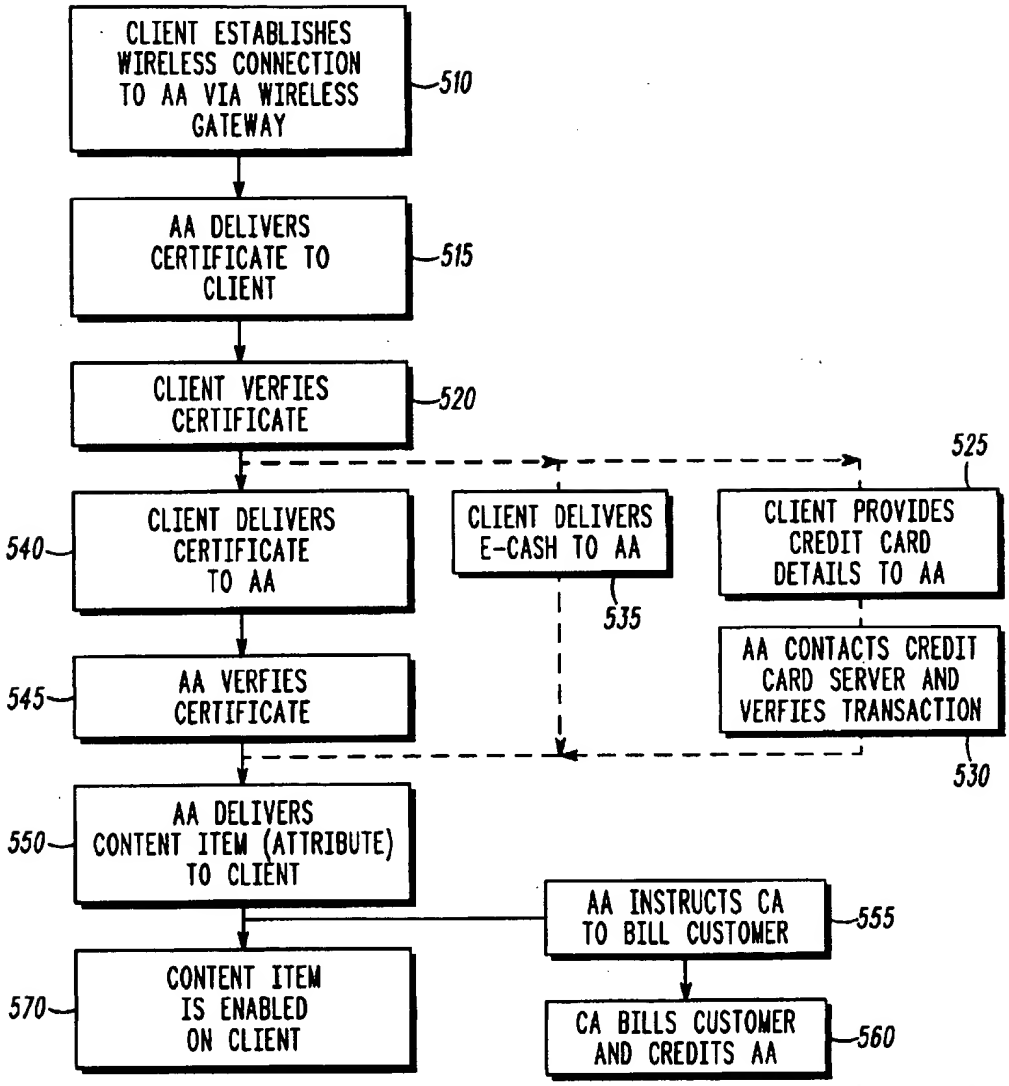

CLIENT ESTABLISHES WIRELESS CONNECTION TO AA VIA WIRELESS GATEWAY
510
AA DELIVERS CERTIFICATE TO CLIENT
515
CLIENT VERFIES CERTIFICATE
520
525
CLIENT DELIVERS CERTIFICATE TO AA
540
CLIENT DELIVERS E-CASH TO AA
535
CLIENT PROVIDES CREDIT CARD DETAILS TO AA
AA CONTACTS CREDIT CARD SERVER AND VERFIES TRANSACTION
530
AA VERFIES CERTIFICATE
545
AA DELIVERS CONTENT ITEM (ATTRIBUTE) TO CLIENT
550
AA INSTRUCTS CA TO BILL CUSTOMER
555
CONTENT ITEM IS ENABLED ON CLIENT
570
CA BILLS CUSTOMER AND CREDITS AA
560

FIG. 5

SECURE WIRELESS ELECTRONIC-COMMERCE SYSTEM WITH DIGITAL PRODUCT CERTIFICATES AND DIGITAL LICENSE CERTIFICATES

FIELD OF THE INVENTION

This invention relates to secure electronic commerce distribution and sales having the ability to offer software enhancements and new features in a simpler, faster, and cheaper method than previously available. Secure electronic commerce brings together three important functions: reprogrammable software or other content (generically referred to also as "product", which includes services); wireless data service; and security (encryption & authentication).

BACKGROUND OF THE INVENTION

Secure electronic commerce offers a way for customers to add or change features in their phone using the convenience of the wireless data service already available in the phone. Moreover, the customer can achieve these goals within minutes and in the comfort of the customer's home or business.

Secure electronic commerce offers many advantages, among them: greater ease of distribution, sale and revenue collection for software-only features; flexible and upgradeable phone platform - this reduces obsolescence; ability to thwart theft of services and cloning; reduced warranty costs in case of software patch updates; and convenience of wireless reprogramming.

SUMMARY OF THE INVENTION

In one aspect, the present invention provides a wireless electronic commerce system comprising a wireless gateway to a wireless network with which a wireless client having a unique client identifier is capable of communicating and at least one server coupleable to the wireless gateway, delivering content items to the wireless device and maintaining digital content certificates for content items and digital license certificates for licenses for the content items. The server maintains, for each wireless client associated with the system, a record of licenses for that client and a record of content items associated with each license.

This record maintained at the server can be used for content item license verification as will be described and is particularly advantageous for repair and recovery as will be described.

In another aspect of the invention, a content item is delivered to a wireless device. The content item is verified with the server. A license associated with the content item is verified and the content item is enabled at the wireless client when the content item and license are verified. A particularly useful (but not essential) feature is that rules can establish that a new license certificate is not required for every new content item.

Glossary of Abbreviations

AA	Attribute Authority
API	Application Programming Interface
CA	Certification Authority
DER	Distinguished Encoding Rules (ASN.1)
EC	Elliptic Curve
GSM	Global System for Mobile Communication

-continued

Glossary of Abbreviations

ID	Identifier
ME	Mobile Equipment
PER	Packed Encoding Rules (ASN.1)
PIN	Personal Identification Number
PK	Public Key
PKI	Public Key Infrastructure
RA	Registration Authority
RSA	RSA (Rivest, Shamir, Adleman) public key algorithm
SHA-1	Secure Hash Algorithm 1
SIM	Subscriber Identity Module
SMS	Short Message Service
WAP	Wireless Application Protocol
WIM	Wireless Identity Module
WML	Wireless Markup Language
WML Script	Wireless Markup LanguageScript
WDP	Wireless Datagram Protocol
WTLS	Wireless Transport Layer Security

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. **1** is a block diagram of a secure wireless electronic commerce system in accordance with a first aspect of the invention.

FIG. **2** is a flow diagram illustrating software installation and boot-up steps for the wireless client device of FIG. **1**.

FIG. **3** is a flow diagram illustrating steps in a process of software download to a wireless device in the field or enabling of software in a wireless device in the field.

FIG. **4** is a block diagram of a secure wireless electronic commerce system in accordance with a second aspect of the invention.

FIG. **5** is a flow diagram illustrating steps of operation of the system of FIG. **4**.

DETAILED DESCRIPTION OF THE DRAWINGS

The overall security model employs cryptographic API and underlying cryptographic toolkits providing a base level of security features, which other stack layers, such as a Wireless Transport Layer Security (WTLS) API, certificate standardization, and wireless applications such as a Wireless Application Protocol (WAP) browser, can build upon. FIG. **1** shows the entities and relationships of the system.

The customer interfaces with the secure electronic commerce system **10** by using a wireless phone **11** or other wireless device plus a landline phone **12** or Internet-accessible computer (dialup, Ethernet, cable, etc.) **13**. Vario within the system **10** perform the tasks of secure electro accessing through an Internet connection **13**, or using a conventional telephone **12** to talk with an operator, the customer inputs an order to the secure electronic commerce system **10** at the Web server **16**.

The web server **16** communicates with the Certificate Authority server **15** to issue a new Product Certificate. This certificate will ensure that only the targeted phone **11** will be able to obtain and use the new feature. Because the phone originally contained a Product Certificate from the Software Server **17** (while in the factory), an audit trail, or accountability, is maintained for the life of the phone. The phone **11** cannot operate software it was not allowed to based on the content of the certificate. The system **10** also contains a copy of the phone's certificate and so it has a record of the capabilities of the phone.

The Certificate Authority server **15** is a server which creates and distributes Public Key Certificates and License Certificates throughout the secure electronic commerce system.

License Certificates allow devices, such as wireless phones, to operate specified software products. License Certificates are issued to each manufactured device prior to leaving the factory, and subsequently when new software is bought. License Certificates the device's serial number as part of the data, which when digitally signed by the CA, will bind the right-to-use software license only to the phone that has that serial number, which by design must be unalterable. Therefore, these certificates can only be used by the targeted party and no one else.

Public Key Certificates enable devices to establish trust through the CA. The CA digitally signs the certificate stating that a given device, denoted by its serial number, has the following public key associated with it.

The Web Server **16** is the front-end server for the secure electronic commerce system. It may be a series of servers (i.e. order entry, billing, order processing, etc.) but conceptually is thought of as one entity. This server contains the order entry system by which customers enter orders. Orders may be taken online through the Web, or by phone via an operator. This server completes the order entry by first verifying the user's information (user name, device serial number, credit card, etc.). The Web server then sends a request to the CA for a new License Certificate. The CA sends the License Certificate to the device. The License Certificate may be "pushed" by the CA or sent on-demand from the device. The device now holds the new License Certificate and has been authorized to use the new software.

The Software Server **17** makes available all of the software products sold by the system provider/operator. It may be a series of servers including various factory servers but conceptually is thought of as one entity. The function of the Software Server **17** is to digitally sign software products and make the software product and corresponding certificate (known as the Product Certificate) available for download.

The function of the WAP Proxy Server **18** is to translate HTML syntax into WML syntax (Internet to WAP protocol) and vice-versa.

Certificates are the cornerstone of the secure electronic commerce system and a description of digital certificates can be found in Draft American National Standard X9.68-199x: Digital Certificates for Mobile, Account Based and High Transaction Volume Financial Systems, available from American Bankers Association Standards Department 1120 Connecticut Avenue., NW Washington DC 20036.

Various of the protocols and techniques in the secure electronic commerce system **10** operate on certificates (either reading/parsing certificates, or adding/modifying/deleting certificates). Every device (servers **16**, **17** and **18** and wireless devices **11**) in the system has one or more certificates. They all have a trusted root certificate which is the Public Key Certificate of the CA from the CA server **15**. Given this, the secure electronic commerce system can be deployed.

There now follows a description of the basic certificate types and the use of certificates in the lifetime of a phone.

A Public Key Certificate contains information that ties a wireless device **11** with its public key. This is accomplished by hashing the relevant data. The certificate is comprised of the aforementioned data plus the hash result. Anyone wishing to verify whether the public key belongs to the device only needs to hash the data again, and verify that it matches the hash result stored with the certificate.

Additionally, the hash is digitally signed by a certifying authority. In this model the CA is the trusted certifying authority. That is, the previously mentioned hash result is signed with the CA's private key. Anyone who has the public key for the CA will be able to verify the encrypted hash. Subsequently, if the verified hash result matches that of the user-computed hash of the certificate data, that tells the user that (i) the certificate must have been signed by the trusted CA since the CA's public key was able to properly verify the signed hash and (ii) the certificate does belong to the subject because the subject's private key can verify data signed by the public key.

A License Certificate contains information that ties a wireless device **11** with certain access rights. In particular, a License Certificate contains, at a minimum, fields for software product and device serial number. The software product field contains a product identifier. This identifier grants the device a license to use the product. A device will be able to run the specified software product if its internal serial number, embedded in the device, matches the License Certificate's serial number. As in the Public Key Certificate, the data in the License Certificate is hashed and signed by the CA. The device will not be able to verify forged License Certificates since it won't be able to validate the certificate to the CA's signature (unless the CA has been compromised).

A Product Certificate ties a content item or subject (e.g. a software product name) to a fingerprint. In this case, the fingerprint is the hash of the software product. Therefore, anyone who has a software product along with its Product Certificate can verify the integrity of the software by comparing a user-computed hash of the software with the hash result stored in the certificate. As in the Public Key Certificate, the hash result in the Product Certificate is digitally signed by the Motorola CA. So, when the user compares the computed hash with the Product Certificate's hash result, a match implies that the software product is the same which the Motorola CA had digitally signed.

Having described the various infrastructure devices and the role that certificates play in the system, the way in which the secure electronic commerce system is deployed can now be described.

The following sections describe briefly how the system functions using examples that occur for a device. The examples chosen are for a phone, and include what happens when the phone is setup in the factory, what happens when a user turns on and uses the phone, and what happens when a user wants to obtain a new feature.

Each phone in the factory has certain some unique characteristics built-in before the unit is shipped. Physically, the phone must contain: (i) ROM available to run unalterable certificate verification code; (ii) EEPROM available to store certificates (access to the certificates storage area must be restricted) and (iii) an unalterable unique serial number (either in ROM, laser-etched, write-once memory, etc.)

At the factory, a Public Key Certificate is generated for the phone (step **100** of FIG. **2**). This can be generated by the Software Server **17**, the CA server **15**, or by the phone itself. The CA server **15** digitally signs the certificate and retains a copy of it (step **101**). The phone is installed with its own Public Key Certificate, plus the CA's Public Key Certificate (**102**). (The CA, being the root certifying authority, signs its own certificate.) A key assumption is that inside the factory there is a trusted network. In this environment, the generation of the phone's Public Key Certificate, its signing by the CA, and the certificate and CA's public key transference into the phone are deemed to be secure.

Also at the factory one or more License Certificates are issued (step **104**). The factory sends a request to the CA

server **15** (or the software server **17** under the root authority of the CA server **15**) to sign a License Certificate (step **105**). The factory provides the CA information on which software licenses or products the phone is supposed to have along with the phone's serial number. The License Certificate includes the following information: (i) the CA's identification (the issuer); (ii) the serial number of the phone (the subject); (iii) a list of software products the phone is licensed to run; and (iv) a signed (encrypted with CA's private key) digest (software hash) of the aforementioned components. The license certificate is installed in the phone at step **106**.

The License Certificate may contain multiple licenses in one certificate, or there may be multiple License Certificates with one license per certificate. Both methods are allowed. The CA itself will retain a copy of the phone's License Certificate(s).

Now, the phone enters the software programming phase. The phone must contains various amounts of software - some base version plus some (optional) additional features, depending on what was ordered. The factory must install the correct software package(s) into the phone. The software packages include the software itself plus a Product Certificate (or more generally a content item certificate). The Software Server **17** generates a Product Certificate (step **110**) for each software product under the same root authority as the CA. Both the software and Product Certificate are stored on the Software Server **17**. The Software Server **17** is responsible for managing the software products and making them available for download. The software package and its certificates (i.e. digitally signed software) is installed in step **112**.

The purpose of the certificate is to bind (i.e associate) the software product to a particular name (e.g. software product name and major version number). This association may be in the form of a look-up list of product names associated with the certificate of a product name and a predefined rule identifying permitted new product names (e.g. Browser version 1. x permits all future versions of Browser between version 1.0 and version 2.0). The certificate contains the name of the software along with a hash of the software product. Anyone wishing to validate the integrity of the software can hash the software and compare it to the one found in the Product Certificate. The certificate is signed by the Software Server. The Software Server itself has a Public Key Certificate signed by the CA, so a line of trust is maintained.

The Product Certificate includes the following information: (i) the Software Server's identification (the issuer); (ii) the software products name (the subject); (iii) a hash of the software product; and (iv) a signed (encrypted with Software Server's private key) digest of the above components.

The Software Server **17** retains a copy of the Product Certificates. It should be noted that the CA **16** may also perform the function of the Software Server **17**, in which case the CA will be responsible for all three types of certificates.

The phone leaves the factory installed with: (i) the CA's Public Key Certificate; (ii) the phone's Public Key Certificate; (iii) one or more License Certificates; and (iv) one or more Product Certificates.

A series of steps takes place every time a user turns on the phone. First, the phone's boot software validates all of its Product Certificates (step **130**). That is, a hash is computed for each software product in the phone and compared against the hash stored in the certificate. Also, a line of trust to the CA must be established. Since the Product Certificate was signed by a Software Server, not the CA, the phone will obtain the Software Server's Public Key Certificate from the CA (this should only occur one time after which the phone will store the Software Server's Public Key Certificate in memory). Next, the phone's boot software validates (step **135**) all of its License Certificates. That it is, a comparison between the phone's unalterable serial number and the serial number stored in the License Certificate(s) associated with the products is made. If they match then the phone is allowed to operate the software products identified in the License Certificate. The software products identified in this certificate must match the software products name field in the Product Certificates. In both steps, the phone's boot software compares the digital signature of the certificates with the CA's Public Key Certificate (which was installed in the factory) to ensure that forged certificates were not installed.

In the event that a user wants to modify the phone, such as enabling an option or purchasing a new feature (see Examples below), a new License Certificate is obtained along with the software product itself plus its Product Certificate. The following steps take place. First, the user purchases the feature through the Web Server **16**. Whether by phone or Web access, the user submits the necessary information to the Web Server **16**, including name, address, credit card number, the phone's serial number, and desired product to purchase (step **16**). The Web Server **16** verifies the information and creates an order ticket which is also given to the customer. Assuming the customers credit has been validated, the Web Server **16** sends the phone's serial number along with the name of the software product purchased, to the CA server **15** to obtain a new License Certificate. The License Certificate is made available to the phone to download (step **204**). The user may initiate a sequence to obtain it. The phone validates the new License Certificate. If the software product was installed in the factory but not enabled (no license certificate), it will now be enabled (step **220**). If the software product is not found in the phone, it will send a request to the Software Server to obtain it (step **226**).

As an optional step, the Software Server **17** may be configured to send software products encrypted and only to authenticated users, in order to prevent overloading of the wireless network and to prevent unauthorized users from obtaining the software code (despite the fact that the phone cannot run the software anyway without a valid License Certificate). This is done by establishing a WTLS connection between the Software Server **17** and the phone **11** via the WAP proxy server **18**. Each party has a Public Key Certificate signed by a common CA, so they trust each other's certificates. Using a key exchange algorithm, a secret key is derived and used for encrypting the software product and Product Certificate. The Software Server then sends the software product and Product Certificate over the air to the phone, the phone validates the Product Certificate and the purchase is complete.

Note that in this model, if the download is interrupted or the software is corrupted, the user can retry the download any number of times.

In the event that the phone has to be brought in to a repair shop, there are a number of features in the secure electronic commerce system available to make updates or modifications to the phone a painless one. For example, if for some reason a different phone is issued to the user, the repair shop could transfer all of the contents of the existing phone (software and Product Certificates) into the new one. The new phone will not be able to run the software until a new set of License Certificates is generated for that phone's serial

number. The repair shop requests new License Certificates from the CA using the repair shop's Public Key Certificate as proof to the CA that it is empowered to do so. The CA has on record the License Certificates for the existing phone. New License Certificates are issued for the new phone, and the existing phone's License Certificates are put on a certificate revocation list. If for some reason the repair shop needed to copy software into the new phone and was unable to copy it from the existing phone, the software (and Product Certificate) could still be downloaded from the Software Server as described previously.

By implementing the secure electronic commerce system, new features, updated software, and software patches can be delivered to the phone in a timely efficient manner for the customer. The customer benefits from the ease of performing the tasks, and the almost immediate cycle time to achieve it. Furthermore, the security features built into the system will reduce incidents of theft of service or cloning.

The following examples help clarify what potential solutions are available in a secure electronic commerce environment for some real world situations.

EXAMPLE 1

Acquiring an Application not Currently in the Phone

A user is travelling overseas and wants to have access to a voice activated German/English translator on their phone. The user can purchase the German/English translator software product from the manufacturer or retailer. The user will download the application into the phone and the German/English translator is operated locally on the phone, rather than through the service provider's infrastucture. This is more cost effective if the user plans to use the feature on a regular basis. The user feels a sense of ownership. As long as the user keeps the phone, he/she owns the software. The software operates locally in the phone. The user does not access the feature over the air, thereby eliminating any potential out-of-service conditions, bandwidth or throughput issues, or unexpected infrastructure downtime. The user pays the manufacturer or the retailer for the feature, rather than the service provider.

EXAMPLE 2

Enhanced Phone Feature

A user wants to be able to use a new Web browser now available for the phone, either buyer downloading the software (if the phone did not contain the software when purchased) or by enabling access rights to the feature already stored within the phone. The user connects the web server **16** (or 1-800 number), follows instructions to purchase additional options for the phone, and waits a short time for the software to be downloaded, or enabled, with the new Web browser capability. The user gains instant satisfaction by accessing the feature minutes after the purchase is made. The cost of the feature is lower than it would otherwise be because the overhead to bring the feature to the user via electronic download is low. Because the transaction is authenticated by the certificate authority via the CA server **15**, theft of service is virtually eliminated.

EXAMPLE 3

Software Patch Update

A software patch is issued to the field to be installed in an existing phones (e.g. under warranty for no charge). The user is instructed how to enable the phone to begin a download containing the updated software. Another option is that the service provider can automatically update phones while the phone is in service, without the user aware of the upgrade. The user does not need to physically return the unit to a service shop. The user's phone is updated in a matter of minutes, rather than days spent in a service shop. An instant electronic record of the software download is retained in the system **10**, rather than relying on field service reports. Certain upgrades can be done automatically without the user's consent, thereby eliminating any, service interruptions. The cost is minimal because no service shop is involved in the procedure.

EXAMPLE 4

Metered Service

A user wants to access an audio book in his/her car during the commute to/from work. The user can purchase an audio book from a service provider. Access may be gained by means of a secure data service connection which allows the metered service (either per book or per minute). Among other options, two-way communications allow the user to suspend/resume the transmission. This is a new type of service not achievable without secure electronic commerce.

Thus a wireless electronic commerce system has been described having a wireless gateway **18** to a wireless network **19** with which a wireless client device **11** having a unique client identifier is capable of communicating. At least one server has been described coupleable to the wireless gateway, delivering content items to the wireless device and maintaining digital content certificates for content items and digital license certificates for licenses for the content items. In the preferred embodiment the server **17** delivers the content and the CA server **15** maintains the digital license certificates. The at least one server maintains, for each wireless client associated with the system, a record of licenses for that client and a record of content items associated with each license. In other words, the CA server **15** maintains a database or list correlating wireless client IDs with licenses (or license certificates) for each client ID and content items (e.g. software products) associated with the licenses.

It has been described that the wireless client is able to request digital license certificate verification for a new content item when the new content item is associated with an existing digital license certificate that is associated with the client identifier. Content of a first wireless client with a first identifier is preferably able to be replicated in a second wireless client with a second identifier by reloading the content to the second wireless client; replacing, in the at least one server, a first association between the first identifier and corresponding records of first client licenses and first client content items, with a new association between the second identifier and the corresponding records of first client licenses and first client content items; and verifying, for the second client, the first client licenses and first client content items, whereby the second client is able to assume the functionality of the first client.

Also described is a method of operating a wireless electronic commerce system comprising: maintaining at at least one server, digital content certificates for content items and digital license certificates for licenses associated with the content items; maintaining at the at least one server, for each wireless client associated with the system, a record of licenses for that client and a record of content items asso-

ciated with each license; establishing communication between the at least one server and a wireless client device via a wireless gateway; delivering a content item to the wireless device having a unique identifier within the system; verifying the content item with the at least one server; verifying a license associated with the content item; and enabling the content item at the wireless client when the content item and license are verified.

The license for the content item is preferably verified when a name for the content item is pre-associated with a digital license certificate associated with the unique identifier for the wireless device. Alternatively the name for the content item satisfies a predefined rule of a digital license certificate associated with the unique identifier for the wireless device, for example it falls within a range of permitted version numbers identifying it as an upgrade of a content item pre-associated with the digital license certificate.

The secure electronic commerce system described offers a solution to enabling software sales over wireless networks. The system is a robust, efficient, and user-friendly method to provide e-commerce services for customers.

The system has been described with a Wireless Application Protocol (WAP) server or gateway **18**, but it will be understood that any wireless network gateway can be used, WAP being merely a convenient protocol. Alternative aspects of the system are now described, also in the context of a WAP protocol and it will be understood that other protocols can be used.

Further details of the WAP server and WAP layers of the secure wireless electronic commerce system are now described. A set of WAP protocols in transport, security, transaction, session and application layers are described in the document "Wireless Application Protocol Architecture Specification" [WAPARCH], WAP Forum, 30Apr.1998. WAP security functionality includes the Wireless Transport Layer Security [WAPWTLS]WAP Forum, 30Apr.1998 and application level security, accessible using the Wireless Markup Language Script [WMLScript]. The security provided in WAP can be of various levels. In the simplest case anonymous key exchange is used for creation of an encrypted channel between server and client; in the next level a server provides a certificate mapping back to an entity trusted by the client; and finally the client itself may possess a private key and public key certificate enabling it to identify itself to other entities in the network. The infrastructure and procedures required to enable the trust relationships needed for authentication of servers and clients are described in greater detail here. The term "server" used here is not limited to a dedicated WAP gateway but may include third parties and content/service providers using the WAP protocols. The specifications for the aforementioned and other WAP protocol layers are found at URL: http://www.wapforum.org/.

In addition to the above-mentioned documents, reference can be made to: "Wireless Control Message Protocol Specification", [WAPWCMP] WAP Forum, Apr. 30,1998; WAP Identity Module Specification [WIM], WAP Forum, Mar. 12 1999; Digital Certificates for Mobile, Account Based, and High Transaction Volume Financial Systems [X968], Mar. 1,1999 ANSI draft; "Standard Specifications For Public Key Cryptography", IEEE P1363/D1a (Draft Version 1a) [P1363], Feb. 1998. URL: http://grouper.ieee.org/groups/1363/; PKCS #1: RSA Encryption Standard", version 1.5 [PKCS1], RSA Laboratories, Nov. 1993; PKCS #15: and Cryptographic Token Information Standard" [PKCS 15], working draft version 1.0, RSA Laboratories, Nov. 1998.

To describe further aspects of the electronic commerce system, the following are now described: the security domains; the attributes to be distributed/protected in the system; attribute ownership assigned to domains (note some attributes are owned by more than one domain); and the architecture for the enrollment and authentication of domain members and assignment of attribute.

A "domain", or "security domain", is a public key infrastructure under the control of a single authority and using a defined internal naming scheme, algorithms, and policies. Domain authority flows for a domain root certification authority having a globally unique name. This allows domains to generate agreements and hook together forming a global PKI. An entity that has been enrolled in a domain by the certifying of the public key that the entity owns within the domain is a "domain member". The following are the possible WAP domains, some or all of which will be referred to: manufacturer(s); network; operator(s); wireless service provider(s); content/services providers (e.g., banking domain); trusted third party domains (e.g., an independent certification agent or authority); device owner (fleet operator domain); and device user (personal domain).

An "attribute" is either a characteristic (which can be considered to be a name) or a right (i.e. a permission, for example a permission to access a purchased service). Examples of attributes, are owned objects (e.g. directories & files, hardware and & interfaces) and owned rights/permissions (e.g. make call; establish network connection; send SMS message; read/write/update files & directories; configure device hardware; access network management station.

In order to implement this security infrastructure, the WAP Public Key system **10** enrolls and authenticates WAP domain members and distribute attributes. Note that the word "distribute" includes "distribute to purchaser"; i.e., subscribers may be purchasing attributes such as access to content or services.

The WAP PKI architecture consists of autonomous security domains tied together by cross-certification. Such cross-certification is a part of service roaming agreements between service providers and system operators. Cross certification is the process by which two domain root CA's issue one another cross-certificates; thereby authorizing one another's root certificates (keys). Cross-certificates generally contain the address of one or more inter-domain validation servers and may also contain other information related to the cross-certification agreements. For the wireless industry cross-certification can be similar to creation of roaming agreements. Within a security domain the algorithms, naming scheme, and policies of that domain are determined by the owner of the domain. During the cross-certification procedure, domains agree on inter-operability issues and configure validation servers to allow certificate validation to be performed.

Classes of WAP operation are: class 0: anonymous authentication only; class 1: server authentication only; and class 2: server and client authentication. Class 0 is not of concern here as it does not involve a PKI. Class 1 involves the authentication of servers to clients. These servers may be owned by the wireless service provider or may be third party servers offering their services in the network. By including these third party servers in the WAP PKI the wireless service provider indicates a special status or stamp of approval to these services. Class 2 operation allows a client (mobile subscriber) to enroll as a member in the domain and obtain benefits of being in such a domain. This may involve special

agreements with third parties such as discounts or special services. Domain members may be issued small attribute certificates, tied to their domain private key, that indicate purchased services or special rights being granted.

Attribute certificates are of two types: (i) purchased service or product certificates, which are like monthly subscriptions and do not require special revocation procedures; and (ii) system operations permissions, which are attributes issued to allow users special system configuration rights. These latter may be used by wireless service provider personnel or issued to authorized subscribers.

Above, several domains have been identified. The usage of some of these domains is now described.

The device owner domain is a domain used to set up device users and user profiles/privileges without wireless service provider involvement. This could be done by having the service provider give the device owner a service provider domain identity and attributes to modify their account configuration. Or the device owner could be made an attribute authority in the service provider domain. The device owner does not necessarily need his own domain, although in a large fleet where device owner needs to issue certificates to employees allowing them certain rights it is preferable to have such a domain. The system allows for device owner domains, although they are not expected to be deployed initially as simpler means (using service provider domain) exist for the required functionality.

Manufacturer domain is used for device bootstrap, device OS code upgrades & features.

Wireless service provider domain and/or network operator domain can be used for WAP gateway certification, content provider certification, subscriber feature distribution, subscriber identification (bind to account), WTA scripts, over air system modifications, distribution of certificates to employees to allow system configuration.

Content/service provider domains can be used for security conscious entities (e.g. banks) to run their own domains that cross-certify to operate with service provider domain or enroll wireless users directly.

Note that a user may be a member (have a key certified) in any domain. A domain is used to issue certificates to grant permissions to others in a system; permissions can be controlled on the device in a simpler manner.

Focus of further discussion herein is directed to the service provider, network operator, and manufacturer domains, with particular reference to FIG. **4**.

FIG. **4** illustrates a CA **400** (implemented as a server). A certification authority is an entity that issues/updates/revokes public key bearing certificates in response to authenticated requests from legitimate registration authorities. The CA holds a private key used to sign domain member key bearing certificates. The CA is managed by a wireless service provider or network operator controlling and operating a wireless network via a wireless network controller **401** and generating bills for customers via billing computer **402**.

A plurality of Attribute Authorities **404**, **405** and **406** are shown (three are illustrated, but there may be many). An Attribute Authority is an entity that generates certificates assigning attributes to domain members. Examples might be a mobile telephone manufacturer **404**, a book merchant **405** and a wireless software supplier **406**.

The AAs can communicate with the end user (client) device **450** via a wireless gateway **420**.

The client device **450** preferably (but not necessarily) has a Subscriber Identity Module (SIM) **452**. This is a smart card in a wireless system holding subscriber identity and authentication information. The SIM card can also be used to run applications needing a secure environment. The client device preferably also has a WAP Identity Module (WIM) that provides an interface for service relating to the use of the WAP security layer as well as data storage services. The WIM uses PKCS **15** for object formats. The WIM may be an interface layer to the SIM card **452**, or it may be a SIM or other card with a native WIM interface on it, or a software token on the wireless device **450**.

All items of FIG. **4** belong to a domain **500**, which in this embodiment is a wireless network operator domain. The domain **500** may have a higher registration authority, that is an entity authorized to make requests to issue/revoke/update certificates to a CA or AA. The registration authority can be considered similar to an account manager in function and is responsible for member enrollment and/or attribute assignments. Enrollment is the process by which a user public key is certified in a domain by the issuance of a domain certificate containing this key.

Server **400** is a validation server. A validation servers is a server that is configured to validate certificates for domain members. Domains that cross-certify are expected to provide accessible validation servers that obtain and validate certificate chains. This service is important when there are domains with local naming schemes. Since these schemes may not be understood by an outside domain, it is necessary for the validation service to be provided. A validation server that is configured to communicate with one or more outside domains is an inter-domain validation server. In addition validation servers may provide local domain validation for thin clients that are domain members but do not have the ability to obtain and validate a certificate chain on their

It is possible to have multiple wireless service provider domains stored in the system. If a customer switches providers in a system with a WIM identity card having personalization and key information on it certain issues arise. If the user simply obtains another card from the new provider all the personalization information will be lost. It is therefore good practice, although not essential, that service provider specific information is separated from personalization information not dependent on the service provider. It is also preferable to provide a means for the user to create a WIM software token to be used to initialize a new WIM card.

It is advantageous for certain businesses wishing to sell services or content to mobile subscribers to have their own security domain. This may be required for certain financial services for risk management reasons. The keys and applications for these domains must be separated from other domains in such a way that the owner is confident that no tampering is possible. Although it is possible to store certificates and keys from any domain in the WIM, this separation can be implemented in various ways that need not be described. In addition to stored object separation the device must insure that domain applications cannot be tampered with.

Any domain member may become an attribute authority. Wireless service providers distribute system management attributes to merchants that they bring into their domain who can sell their attributes- i.e. sell a purchased service ticket (e.g. for new software as described above or for other content or services).

There are three types of payloads: compact payloads, organizational payloads of which WAP payloads are one type, and domain specific payloads that are valid only in a local domain.

In order for merchants operating AA servers **404–406** to define simple attributes for the service/content they sell, it is necessary to have a method to indicate that a given attribute payload is a merchant payload and to indicate the particular merchant. Since the payloads are not expected to be understood by anyone but the merchant the actual structure of the payload is not material. There are three ways that merchant attribute payloads may be identified: (i) if the merchant has an organizational identifier it can define its types under this identifier; (ii) merchant identifiers can be assigned under the WAP OID; (iii) domains can defines and deploy a merchant ID scheme for their domain specific payloads.

Attribute certificates are preferably time-limited, i.e have an expiry date and be subject to periodic renewal. At any time when a certificate is verified, verification fails if an expiry time digitally embedded in the certificate has expired. A mobile client checks the validity of a merchant server (**404-406**) with the network upon first use or periodically. The client (either the wireless device **450** or the SIM **452** can be considered a client) contacts the validation server via the wireless gateway **420** for this purpose. In addition the wireless service provider may post a web site listing the status of domain merchants. Such a site could also indicate if the merchant is no longer a trusted domain member; i.e., if their certificate has been revoked for some reason.

The steps by which a merchant (i.e. an AA) or other network element having the capacity of an AA delivers content to a client and receives payment for that content is now described with reference to FIG. **5**.

Initially, a client (**450** or **452**) establishes connection in step **510** with an AA server **404** via the wireless gateway **420**. The AA delivers a digital certificate to the client in step **515**. The client verifies the AA certificate in step **520**. This is achieved at the client in the same manner as other certificate verification already described and is achievable because the client already has installed the public key certificate of the CA **400**. It uses this public key certificate to verify the AA. The reason for this verification is because the client wishes to have confirmation that the AA is trusted by the wireless service provider. This will give the user confidence that the content to be delivered by the AA will operate when delivered to the wireless device **450**, that it will not cause disruption, that the fee to be paid is as advertised, that upgrades will be available, etc. There are many reasons why the user may wish to ensure that the AA is certified by the wireless service provider.

Next, an exchange occurs between the client and the AA to deliver payment to the AA for the content that the client is about to receive. This transaction can take one of several forms. The AA informs the client what the fee is for the content to be received and the client, if the user so chooses, authorizes payment of that fee. For example, the client can provide credit card details (step **535**) and the AA server can establish a connection with a credit card server to execute a transaction (step **530**). Alternatively, the client can deliver electronic cash to the AA server (step **535** (which need not be described). In a more preferred alternative arrangement, the client delivers a certificate to the AA that is certified within the wireless service provider domain **500**, i.e. is certified by the CA **400** and has the public key certificate of the CA **400** (step **540**). The AA server verifies this certificate in step **545** (using the root public key certificate that it already has stored at the AA server) and the AA is ready to deliver the content item (attribute) to the client. Upon delivery of the content item to the client (step **550**) the AA instructs the CA to bill the customer for the predetermined fee (step **555**) and the CA **400** instructs the billing computer **401** to add the fee to the customer's bill (step **560**). Meanwhile (step **570**) the content item is enabled on the client.

Enabling of the content item can take many forms, for example executing a software application or an upgrade or patch to a software application, or displaying a newspaper, or delivering a stock quote service, or opening wireless web access to a streaming video service or music service, or delivering an electronic airline ticket or a ticket to a concert, or many other examples. In effect, the various AAs **404, 405, 406** all provide a virtual wireless shopping mall with a common billing mechanism, which is the billing mechanism of the wireless service provider.

If steps **535** or **525** are executed, it is not strictly necessary for the AA to perform online certificate validation of the client's certificate (steps **540** and **545**), but is preferable to do so anyway. As an alternative to the step of verifying the AA certificate (step **520**) the client can just check the last posted AA list (or AA revocation list). In addition the domain provides a service that notifies merchants when a client certificate is revoked, so as an alternative to step **545**, the AA can check this list. Of course, if the client is charged by electronic cash or credit card, steps **555** and **560** are omitted.

Although several domains may be useful, it is not necessary to have users become members of these domains by obtaining a user key. In the case of a phone, as already described, the manufacturer can enroll users if direct sales to a user of device feature upgrades are intended. This could, however, also be done through distribution through the service provider domain or assignment to device identifiers. Managing user keys may be less preferable in the manufacturer domain. This domain can preferably be used to authenticate features and upgrade scripts from the manufacturer.

If the network operator runs a security domain, client keys may be required, but only for clients that are to be authorized to perform network operations from the mobile device (i.e., network operator employees). General client authentication to the network operator is not strictly necessary.

The case of the wireless service provider assigning keys to users is of particular interest. The client keys are distributed in the WIM card when service is purchased. The client key in the wireless service provider domain allows a member of this domain to assert an identity that is bound to the account with the provider. The service provider or its authorized third parties may then sell content/services in the domain. The client key also allows certain clients (say, service provider employees) to use the system to identify themselves to the network as having network operations permissions.

For service and content providers other than the wireless provider that require separate security domains client keys are preferably issued and stored in the WIM. Domains are permitted to specify any naming scheme consistent with their network and are responsible for assuring that this scheme is internally consistent. A domain's internal directory service must be configured to resolve local domain names. Names may be of multiple type including email addresses, IP addresses, DNS names, and account numbers. The business deploying (or contracting for deployment of) a domain may configure the naming mechanism to suit their business and system needs.

The AAs **404, 405** and **406** can communicate between each other via links **408** and **409** to exchange electronic vouchers. For example, AA **404** can advertise to its clients that purchase of its services earns for the client a credit for the services offered by AA **405**. When a client purchases a

content item from AA **404**, AA **404** generates an automatic voucher that it delivers to AA **405**, identifying the client by ID and the credit to be attributed to that client. When the client establishes a connection to AA **405**, the client is informed of the discount from the fee of AA **405** that this particular client has earned. If the client purchases a service from AA **405**, the latter can charge AA **404** for the value of the discount voucher or a portion thereof. There are many variations on this scheme that will be readily apparent to one of ordinary skill in the art. For example, points can be accumulated by the billing computer at the CA **400** that can be used at any of the AAs **404–406**. Alternatively, the voucher can be transferred to the client **450** or **452** instead of between the AAs so that the client can deliver the voucher to the next AA in the virtual wireless mall. This latter scheme is particularly simple to implement using the already describes public key certificate common to all the members of the domain **500**, because any AA in the domain can readily verify the authentiity of the voucher.

Further miscellaneous details of the overall secure wireless electronic commerce system are now given for completeness.

Directories are organized as: domain authority name->fully qualified local descriptive name->key hash, certificate hash, or key ID or domain authority name->key hash, certificate hash. Within a domain the domain authority name is implicit. A fully qualified local name is a local name that is unique within the domain; in some cases this is a combination of the issuer and subject names in a certificate (subject relative to issuer). The use of the certificate or key hash alone (no descriptive name part) for entity names is less preferable due to the difficulties it creates in directory lookup and certificate chain validation.

An issuer below root has a name consisting of either a certificate hash and an optional descriptive name, a key hash and an optional descriptive name, or an integer key identifier and a descriptive name. Use of the integer key identifier and descriptive name allows for shorter and more human readable names.

The validator has means to obtain the necessary certificate path when a transaction requires that a certificate chain be processed. In many cases the end entity certificate itself will not be sent, only a name or names that allow the validator to obtain any required certificates. Within a domain this can be made simple by using a deterministic path naming scheme that allows the certificate validation path to be obtained directly from the (issuer, subject) pair. If the naming scheme does not do this internal to a certificate then a means external to the certificate must be provided to obtain the proper path. For validation between cross-certified domains it is recommended that a domain provide a secure validation service for its certificates. If a domain is to validate certificates from other domains internally then it must understand the naming schemes and algorithms used in the other domains.

In certain cases a CA may have more than one valid certificate listed at a given time. This could happen during a scheduled re-key of the CA. This creates a possible problem when trying to follow a certificate chain as at some point the chain allows multiple possibilities. There are several methods for dealing with this: (i) trial and error, which is inefficient due to the need to check multiple signatures for a match; (ii) use of a name type containing a key or certificate hash for CA's and AA's. This resolves ambiguity but makes names larger; (iii) use a key identifier with a descriptive name . This latter is the preferred method.

Given that some devices may not be capable of processing certificate chains for intra-domain validation, a validation service is defined in X9.68. This service is realized by providing validation servers in a domain. The addresses of these may be indicated as payloads in CA certificates or configured in some other manner. The client of the service must be able to verify the signature on the response so it must have the validation server certificate or be able to verify this certificate. A thin client receives a certificate that it wishes to validate along with an indication of the domain authority of the certificate; the domain is its own. The client sends a validation request to a validation server. The server sends back a signed validation response.

For inter-domain validation, domains make certain services available to other domains. A cross-certificate indicates where these services may be obtained by providing an address for servers offering such services. The validation service operates between a validation client, which may be an intra-domain validation server, and an inter-domain validation server. An inter-domain validation server is expected to be able to contact and verify the authenticity (signature) of a response from another domain's validation server(s). Note that this implies that inter-domain validation servers must be capable of using the algorithms from the other domain. Inter-domain validation servers are members of multiple domains and therefore are able to verify signatures of other domains. A thin client receives a certificate that it wishes to validate along with the domain authority of the certificate; the certificate domain is not its own. The client sends a validation request to a domain validation server. Note these servers can be indicated in the root certificate as payloads or stored elsewhere. The validation server notes that the domain is external. It determines if a cross-certificate with the indicated domain exists. If not a validation failure response is returned to the client. (Note that this third step can be skipped if the client is aware of the address of an inter-domain validation server). If a cross-certificate exists and the server is an inter-domain validation server it handles the request itself, otherwise it signs and sends a validation request to an inter-domain validation server listed in the cross certificate. The inter-domain validation server authenticates the request and either verifies the certificate itself or passes the request to an internal validation server in the other domain. The result is signed and returned to the validation server the client originally contacted. The validation server authenticates the response, signs it itself, and returns it to the client.

The validation service allows an entity from one domain to request that a certificate from another domain be authenticated. A reply of invalid is required to be returned unless the server can ascertain that the certificate is currently valid. A reason for the failure is preferably provided.

Each domain selects the algorithms to be used internally for digital signatures, data encipherment, key encipherment, and key agreement. In addition the maximum key size that can be used inside the domain is given by the domain root CA key size. Domain members must have the cryptographic software/hardware implementing the domain algorithms with the maximum key size. In addition a domain member must have a securely loaded and stored domain root CA certificate or a condensed form of the domain root CA certificate information.

A domain is defined so as to fix the algorithms used within for efficiency reasons. As a domain selects RSA with a given root key size the algorithm and key size are known for every client in that domain. This makes the impact of supporting another algorithm for a mobile client explicit- every new

domain the client joins is potentially a new algorithm. In fact, for security reasons, some domains may not even allow another domains code to be used even if the algorithm is the same.

A domain root CA indicates the algorithms, internal naming scheme, and policies in effect in the domain. Policies include such information on what guarantee the domain makes of member identity binding to account and what procedures and guarantees are made for third parties issuing attribute certificates in the domain. In addition specific policies for each cross-certification agreement are maintained.

The process of cross-certification is one in which two autonomous security grant formal recognition to one another. This is done digitally by each domain creating and signing a cross-certificate object containing the other domains root certificate (or its hash). In addition the addresses of inter-domain validation servers and indications of contractual agreements may be contained.

For service provider domains in systems with SIM or WIM smart cards that are personalized at the time of service subscription, member enrollment comprises storing the member certificate and key on the WIM card.

For systems not having a smart card, the keys and certificate are stored in a WIM software token. Member keys are protected by passphrase information. This information is concatenated with a stored secret value for the device and run through a secure hash in order to generate the encryption/decryption key for use in protecting the user private key. Member certificates are also kept with the member's account and entered into the service provider member directory. In some systems the actual member certificate may not be stored on the WIM itself but maintained in the system only. For this usage the member certificate can be used as a secure container for account information and a client need only to present his identity to the system; the system obtains the required certificate. For domains other than the service provider the enrollment process requires that a secure manner to store the keys in the WIM be defined and that each domain is assured of information

System bootstrap involves the loading of the domain root certificates and client keys. Note that in addition to a client signature key, keys to be used for data encipherment, key encipherment, and key agreement can also be loaded. Signature keys for clients must be used only for signatures. Keys for the other purposes are also loaded. Initially the manufacturer domain root certificate is loaded into the mobile device. This certificate is used to enable secure loading of other domain root certificates if necessary. In systems not having a secure service provider provisioning scheme in place, the manufacturers provide certificates to service providers giving the service provider a provisioning attribute. This attribute allows a service provider to install their root certificate in the phone.

It is necessary to securely load and store domain certificates. Initially the device manufacturer certificate is loaded into the device as indicated above. Next service provider and network operator certificates are loaded. Once valid service provider and network operator certificates are loaded these domains gain control of attributes their domains own. Manufacturer domain loses control of any attributes it does not own in a provisioned device.

For systems with SIM cards, the wireless service provider root certificate and client key and certificate are distributed on the card before it is issued to the subscriber. The means used to generate and bind the client key insure that only one binding is possible for a card and that signature keys cannot be determined by the service provider or their employees. The personalization process binds the public key (and hence the private key) for a card to an individual account.

For systems without identity cards, the service provider and network operator certificates are loaded using the manufacturer root certificate to secure the loading process. The manufacturer issues certificates to service providers allowing them to sign key loading scripts. The device must insure that only a properly authenticated request be used to load a domain root certificate.

During a planned re-key, each CA or AA simply re-issues certificates. A device must be able to accept a certificate change order for a certificate signed by the owner of that certificate.

When a CA or AA key other than root is compromised, messages from root can instruct the WIM to replace the key with a new key. Using root for this purpose means lower level CA's need not have their own disaster recovery keys, only root requires a key for this purpose.

When a root key is compromised it is desirable to have procedures in place allowing recovery without re-enrollment of all domain members. For this reason each domain preferably has a disaster recovery public key for a domain securely stored in the WIM along with the domain root public key. The private disaster recovery key must be kept in a separate system that is not connected to any network unless disaster recovery is underway. It is desirable that the location for this key be physically separate from the domain root key. Disaster recovery messages are messages instructing the WIM to automatically replace a root key. It is required that both the current root key and the disaster recovery key sign the root public key change message. Receipt of a request for which one signature checks but the other is invalid must cause the WIM to be disabled. It is also required that a message disabling the SIM/WIM card entirely, signed by the disaster recovery key, be supported. In summary, the disaster recovery process is: message to change root signed by current root key and disaster key; (ii) message to fully disable WIM may be signed by either root or disaster recovery key; and (iii) receipt of a partially invalid request must cause WIM to be disabled.

A member device has keys associated with it that may be used for transactions or authentication. These keys are generally protected by a PIN code and some "maximum number of tries" deactivation protocol. It is desirable that when a device is reported stolen, and the report itself has been authenticated in some manner, the system be able to cause a card to clear keys remotely if someone attempts to use the stolen device.

Thus a method of conducting transactions in a wireless electronic commerce system has been described, where the system comprises a wireless network operator certification authority having a root public key certificate and at least one attribute authority having a digital certificate that is dependent from the root public key certificate, where the attribute authority is accessible by a wireless client device via a wireless network. According to the method described, a wireless communication is established between the wireless client device (**450** or **452**) and the attribute authority (**404**, **405** or **405**). The digital certificate is delivered from the attribute authority to the wireless device, the attribute authority is verified to the wireless client device using the digital certificate and the root public key certificate preloaded in the wireless client device under authority of the

wireless network operator. An attribute is delivered to the wireless client device over the wireless network and ultimately enabled at the wireless client device.

Payment for the attribute may be transacted by delivering a second digital certificate from the wireless client device to the attribute authority and verifying the second digital certificate using the root public key certificate from the certification authority. An instruction is preferably transferred from the attribute authority to a billing computer of the wireless network operator to add an item to a wireless network operator bill for the wireless client device.

The second digital certificate may be pre-loaded into the subscriber identity module by the wireless network operator or pre-loaded into a wireless communicator under authority of the wireless network operator.

Also described is a method of conducting transactions in a wireless electronic commerce system comprising a wireless network operator certification authority having a root public key certificate and at least first and second attribute authorities, having respective first and second digital certificates that are dependent from the root public key certificate, where the attribute authorities are accessible by a wireless client device via a wireless network. The method includes establishing a wireless communication between the wireless client device and the first attribute authority; delivering a first attribute to the wireless client device over the wireless network; generating an electronic voucher verifiable by the second attribute authority; establishing a wireless communication between the wireless client device and the second attribute authority; requesting a second attribute from the second attribute authority; identifying the electronic voucher at the second attribute authority; and delivering the second attribute from the second attribute authority to the wireless device. The step of generating an electronic voucher verifiable by the second attribute authority may include delivering the electronic voucher from the first attribute authority to the second attribute authority via a connection therebetween or may include delivering the electronic voucher from the first attribute authority to the second attribute authority via the wireless client.

Also described is a wireless electronic commerce system comprising: a wireless network operator certification authority server having a root public key certificate; at least one attribute authority server coupleable to the wireless network operator certification authority server, having a digital certificate that is dependent from the root public key certificate; a wireless client device having pre-loaded therein the root public key certificate; a wireless network coupling the wireless client device to the at least one attribute authority server; verification means in the wireless client device for verifying the digital certificate to the wireless client device using the root public key certificate pre-loaded in the wireless client device; means (e.g. server **18** or gateway **420**) associated with the attribute authority server for sending and means (e.g. a radio receiver in the wireless device **450**) for receiving an attribute over the wireless network; and means for enabling the attribute at the wireless client device.

The above description has been given by way of example only. Other aspects, objects and advantages of the invention and modifications of detail will be apparent to one of ordinary skill in the art reading the teaching herein. All such modifications are included within the scope and spirit of the claims.

What is claimed is:

1. A wireless electronic commerce system comprising:

a wireless gateway to a wireless network with which a wireless client device having a unique client identifier is capable of communicating; and

at least one server coupleable to the wireless gateway, delivering content items to the wireless client device and maintaining digital content certificates for content items, the digital content certificates establishing that the content items are available on the wireless client device and digital license certificates for licenses for the content items to enable only a specified device to operate a specified content item such that the digital license certificate can be used only by the specified device and no other device,

wherein the at least one server maintains, for each wireless client device associated with the system, a record of the licenses for the content items available on that wireless client device and a record of the content items associated with each of the licenses, the record of the licenses and the record of the content items enabling content item license verification on the wireless client device.

2. The wireless electronic commerce system of claim 1, wherein a wireless client device is able to request digital license certificate verification for a new content item when the new content item is associated with an existing digital license certificate that is associated with the client identifier.

3. The wireless electronic commerce system of claim 1, wherein content of a first wireless client device with a first identifier is able to be replicated in a second wireless client device with a second identifier by:

a reloading the content to the second wireless client device;

replacing, in the at least one server, a first association between the first identifier and corresponding records of first client licenses and first client content items, with a new association between the second identifier and the corresponding records of first client licenses and first client content items; and

verifying, for the second wireless client device, the first client licenses and first client content items, whereby the second wireless client device is able to assume the functionality of the first wireless client device.

4. A method of operating a wireless electronic commerce system, comprising:

maintaining at least one server, digital content certificates for content items and digital license certificates for licenses associated with the content items, the digital content certificates establishing that the content items are available on the wireless client device and the digital license certificates to enable only a specified device to operate a specified content item such that the digital license certificate can be used only by the specified device and no other device;

maintaining at the at least one server, for each wireless client device associated with the system, a record of licenses for that wireless client device and a record of content items associated with each license, the record of the licenses and the record of the content items enabling content item license verification on each wireless client device;

establishing communication between the at least one server and a wireless client device via a wireless gateway;

delivering a content item to the wireless client device having a unique identifier within the system;

verifying the content item with the at least one server;

verifying a license associated with the content item;

enabling the content item at the wireless client device when the content item and license are verified.

5. The method of claim 4, wherein the license for the content item is verified when a name for the content item is pre-associated with a digital license certificate associated with the unique identifier for the wireless client device.

6. The method of claim 4, wherein the license for the content item is verified when a name for the content item satisfies a predefined rule of a digital license certificate associated with the unique identifier for the wireless client device.

7. The method of claim 6, wherein the predefined rule includes determining whether the name of content item identifies it as an upgrade of a content item pre-associated with the digital license certificate.

* * * * *

US006662167B1

(12) **United States Patent**
Xiao

(10) **Patent No.: US 6,662,167 B1**
(45) **Date of Patent: Dec. 9, 2003**

(54) **METHOD FOR GENERATING NEAR-OPTIMAL SEQUENCING OF MANUFACTURING TASKS SUBJECT TO USER-GIVEN HARD AND SOFT CONSTRAINTS**

(76) Inventor: **Jing Xiao**, 4716 Heatherton Pl., Charlotte, NC (US) 28270

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

(21) Appl. No.: **09/461,962**

(22) Filed: **Dec. 15, 1999**

Related U.S. Application Data

(60) Provisional application No. 60/112,329, filed on Dec. 15, 1998.

(51) **Int. Cl.**[7] **G06N 3/00**; G06N 3/12; G06F 15/18

(52) **U.S. Cl.** **706/13**; 706/12; 706/912

(58) **Field of Search** 706/13, 12, 912

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,319,781 A	6/1994	Syswerda	705/8
5,467,268 A	11/1995	Sisley et al.	705/9
5,541,848 A	7/1996	McCormack et al.	700/213
5,727,130 A	3/1998	Hung	706/13
5,761,381 A	6/1998	Arci et al.	706/13
5,764,953 A	6/1998	Collins et al.	703/17
5,787,283 A	7/1998	Chin et al.	717/101
5,839,120 A	11/1998	Thearling	706/13

OTHER PUBLICATIONS

V. Tam; An efficient Heuristic-Based Evolutionary Algorithm for Solving Constraint Satisfaction Problems; May 21-23, 1998; IEEE: 0-8186-8548-4.*

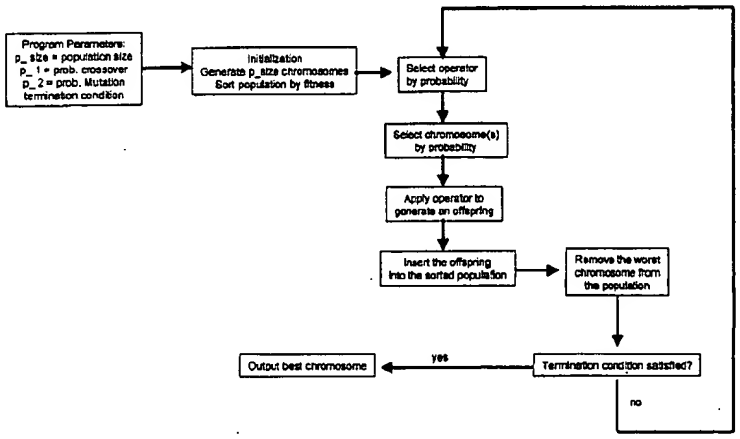
(List continued on next page.)

Primary Examiner—Wilbert L. Starks, Jr.
Assistant Examiner—Joseph P. Hirl
(74) *Attorney, Agent, or Firm*—Kilpatrick Stockton LLP

(57) **ABSTRACT**

The present invention provides a method utilizing evolutionary processes for solving partial constraint satisfaction problems in order to produce a near-optimal or optimal sequence of products for manufacture. More specifically, a computer implemented method for generating an optimized sequence of "N" number of products for manufacture is provided, where said products are of "M" number of distinct types with a fixed number ("N_i") of each type being desired and each product type comprising an array ("Q") of distinct features, wherein said manufacture is optionally constrained by one or more of the following constraints: the production requirement for each product type, feature-based position equations, and feature-based position inequalities, wherein each of said constraints is individually designated as either a hard constraint which cannot be violated, or as a soft constraint which can be violated at a predetermined cost; said method comprising: generating an initial population of chromosomes, wherein each chromosome represents a feasible sequence of products of various types for manufacture, feasibility depending on satisfaction of all of said hard constraints; associating a fitness value with each chromosome, said fitness value being a function of the predetermined cost associated with the degree of violation of each of said soft constraints; sorting said chromosomes based on the fitness value associated with each chromosome; and applying iteratively to the population of chromosomes a reproductive process, comprising (1) selection of a genetic operator, (2) selection of one or two chromosomes, the number of chromosomes to be selected correlating with the selected genetic operator, (3) application of the selected genetic operator to the selected one or two chromosomes to cause generation of one or two offspring, (4) insertion of one offspring chromosome into the sorted population, and (5) discard of one of the least desirable chromosomes in the population; said iterative process being continuously run until the fitness value for the best chromosome satisfies a known criterion or until a pre-determined time has elapsed.

26 Claims, 5 Drawing Sheets



OTHER PUBLICATIONS

Reeves. Colin R., et al., "Genetic Algorithms, Path Relinking, and the Flowshop Sequencing Problem," *Evolutionary Computation*, vol. 6, No. 1, Spring 1998, pp. 45–60.

Warick, Terry et al., "Tackling Car Sequencing Problems Using a Generic Genetic Algorithm," *Evolutionary Computation*, vol. 3, No. 1, 1996, pp. 267–298.

Back, T., Fogel, D.B. and Michalewicz, Z, eds.,*Handbook of Evolutionary Computation*, New York: Oxford Univ. Press and Institute of Physics, 1997.

Bellman, R.E. and Dreyfus, S.E., *Applied dynamic programming*, Princeton University Press, 1962.

Burke, Edmund K., et al., "Initialization Strategies and Diversity in Evolutionary Timetabling," *Evolutionary Computation*, vol. 6, No. 1, Spring 1998, pp. 81–103.

Cheng, J., Y. Lu, G. Puskorius, S. Bergeon, and J. Xiao, "Vehicle Sequencing based on Eovlutionary Computation," *Proceedings of 1999 Congress on Evolutionary Computation*, Washington D. C., Jul. 6–9, 1999, pp. 1207–1214, IEEE Press.

Cotta, Carlos et al., "Genetic Forma Recombination in Permutation Flowshop Problems," *Evolutionary Computation*, vol. 6, No. 1, Spring 1998, pp. 25–44.

Deb, Kalyanmoy, "Time Scheduling of Transit Systems With Transfer Consideration Using Genetic Algorithms," *Evolutionary Computation*, vol. 6, No. 1, Spring 1998, pp. 1–24.

Fogel, Lawrence et al., "Artificial Intelligence Through Simulated Evolution," John Wiley & Sons, Inc., 1966.

Garey, M. and Johnson, D. , *Computers and Intractability: a guide to the theory of NP–completeness*, Freeman, 1979.

Hart, Emma et al., "Solving a Real–World Problem Using an Evolving Heuristically Driven Schedule Builder," *Evolutionary Computation*, vol. 6, No. 1, Spring 1998, pp. 61–80.

Holland, John H. *"Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence,"* The University of Michigan Press, 1975.

Lawler, E. L. and Wood, D. E., "Branch and bounds methods: A Survey," *Operations Research*, 14, 699–719, 1986.

Michalewicz, Z. *Genetic algorithms+ data structures=evolution programs*, 3rd ed., Springer–Verlag, 1996.

Mitchell, Melanie "An Introduction to Genetic Algorithms," The MIT Press, 1996.

Montana, David, "Introduction to the Special Issue: Evolutionary Algorithms for Scheduling," *Evolutionary Computation*, vol. 6, No. 1, Spring 1998, pp. v–ix.

Sebaaly, M.F. and Fujimoto, H., *"A Genetic Planner for Assembly Automation,"* Proceedings of IEEE International Conference on Evolutionary Computation, V20–22, May 1996, pp. 401–4066.

Srivans, M. et al., *"Adaptive Probabilities of Crossover and Mutation in Genetic Algoritms,"* IEEE Transactions on Systems, Man and Cybernetic, IEEE Inc. New York, vol. 24, No. 4, Apr, 1, 1994, pp. 656–667.

Syswerda, G. et al., *"The Application of Genetic Algorithms to Resource Scheduling,"* Proceedings of the International Conference on Genetic Algorithm, US, San Mateo, Morgan, Kaufman, vol. Conf. 4, 19991, pp. 502–508.

* cited by examiner

Figure 1

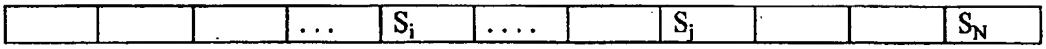


Figure 2

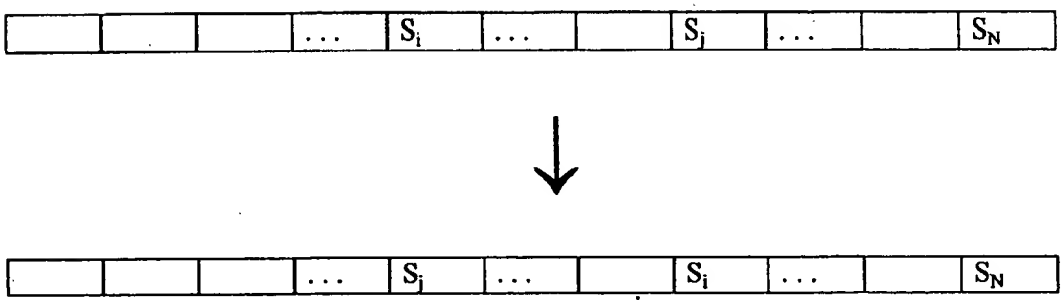
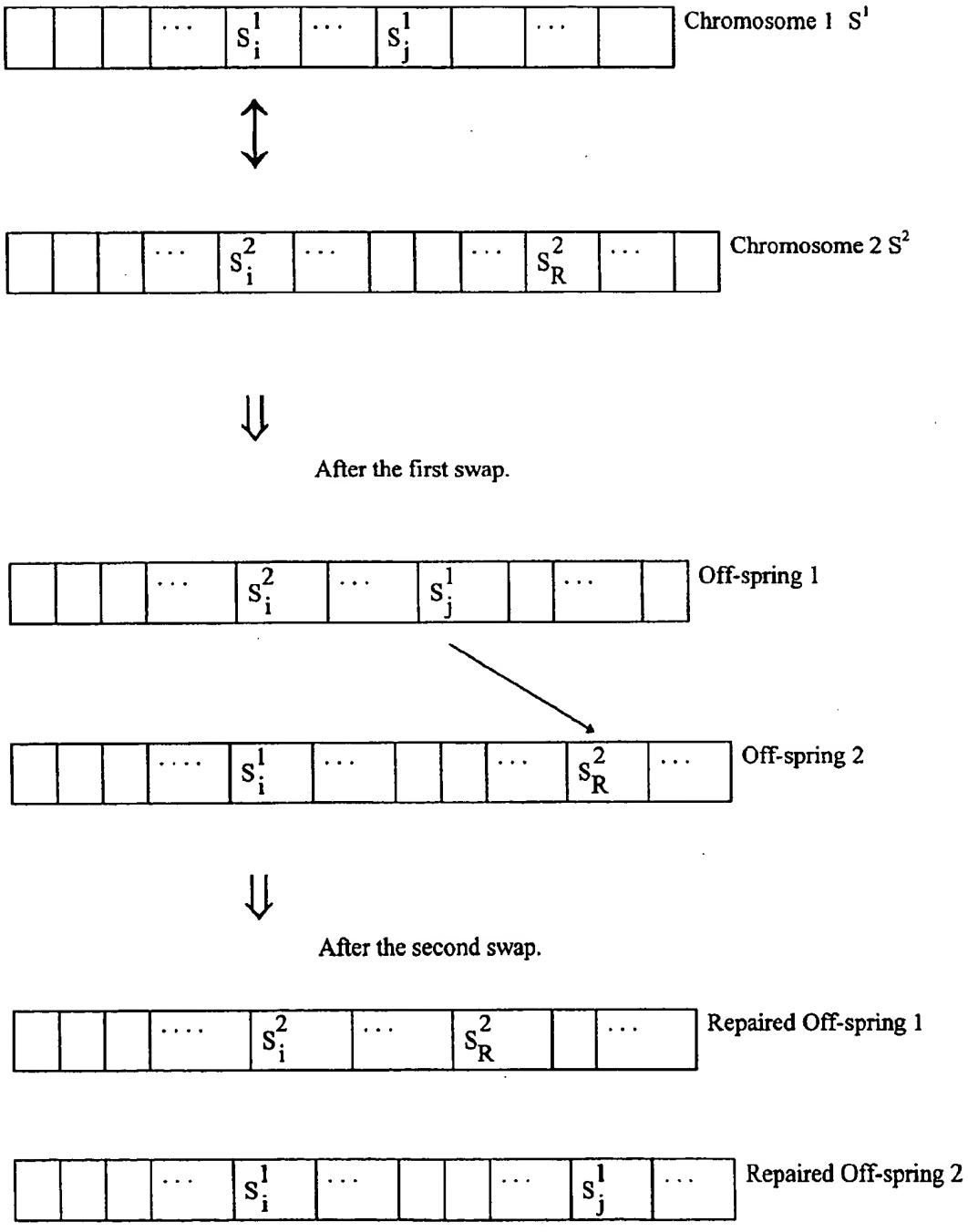


Figure 3


Chromosome 1 S^1
Chromosome 2 S^2
After the first swap.
Off-spring 1
Off-spring 2
After the second swap.
Repaired Off-spring 1
Repaired Off-spring 2

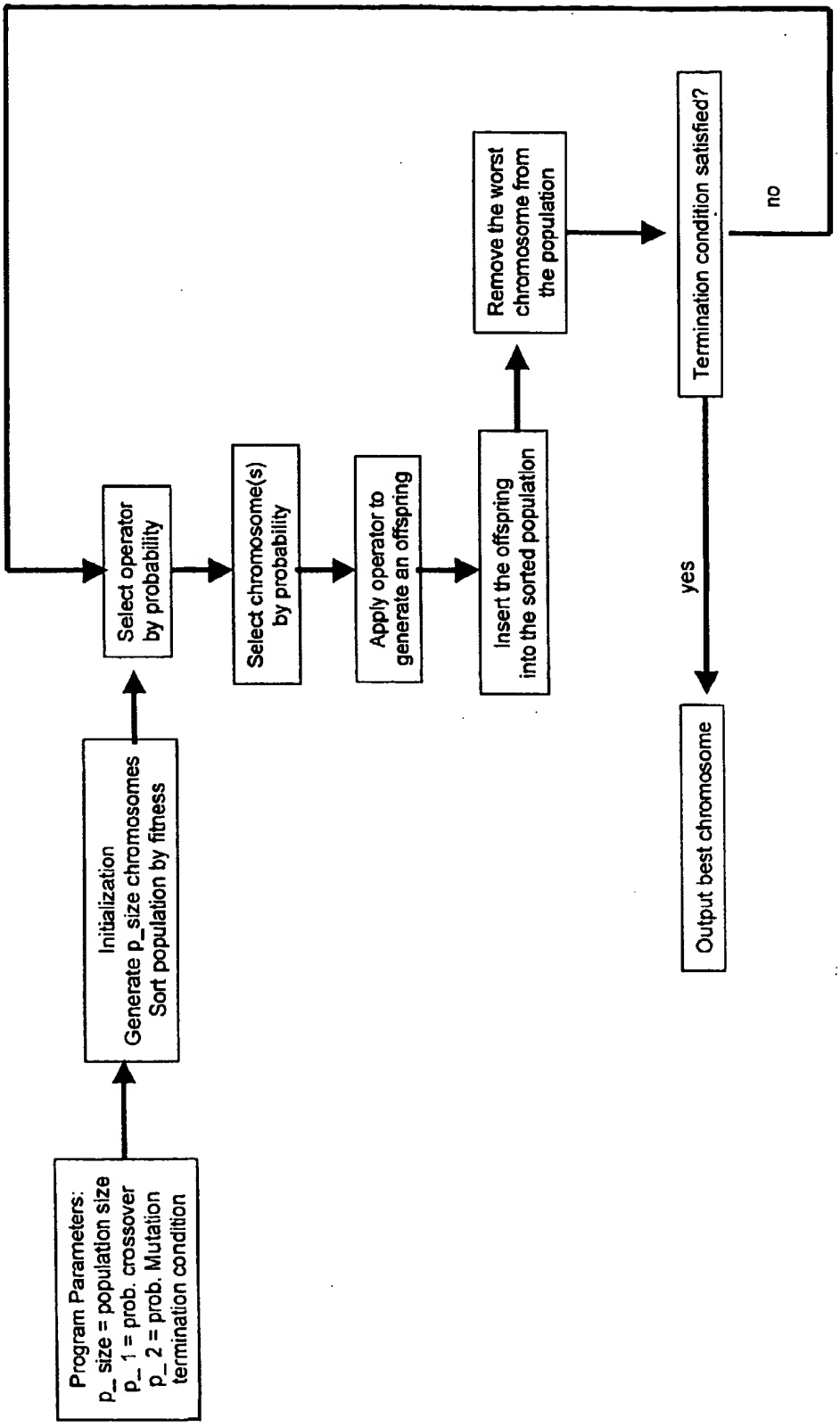
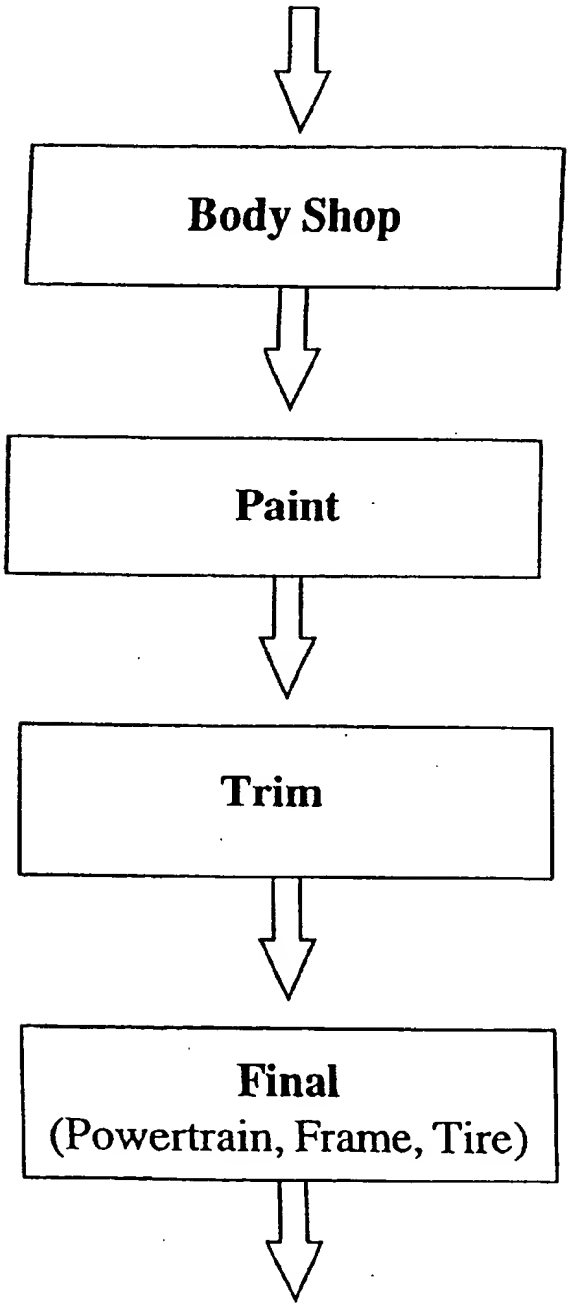

Program Parameters:
p_ size = population size
p_ 1 = prob. crossover
p_ 2 = prob. Mutation
termination condition
Initialization
Generate p_size chromosomes
Sort population by fitness
Select operator by probability
Select chromosome(s) by probability
Apply operator to generate an offspring
Insert the offspring into the sorted population
Remove the worst chromosome from the population
Termination condition satisfied?
no
yes
Output best chromosome

Figure 4

Figure 5

VEHICLE ASSEMBLY SEQUENCE DIAGRAM



METHOD FOR GENERATING NEAR-OPTIMAL SEQUENCING OF MANUFACTURING TASKS SUBJECT TO USER-GIVEN HARD AND SOFT CONSTRAINTS

This application claims the benefit of Provisional Application No. 60/112,329 filed, Dec. 15, 1998.

FIELD OF THE INVENTION

The field of the invention is evolutionary processes; more particularly, the present invention relates to a method utilizing evolutionary processes for solving partial constraint satisfaction problems in order to produce a near-optimal or optimal sequence of products for manufacture.

SUMMARY OF THE INVENTION

The present invention relates to a method for solving partial constraint satisfaction problems, more particularly to a method of sequencing products for manufacture such that the sequence produced is a near-optimal or optimal sequence, meaning that direct cost associated with various labor, processes, and parts inventory are minimized and equipment and floor space utilization are maximized. One application of the method of the present invention is the sequencing of automobiles during manufacturing, in which vehicles are sequenced to go through a series of operations such as body forming, painting, component assembly (such as installing radios, seats, etc.) and final assembly (adding trim and chassis).

BACKGROUND OF THE INVENTION

The sequencing of manufacturing stages or tasks represents a combinatorial, or partial, constraint satisfaction and optimization problem.

As is understood by those skilled in the art, partial constraint satisfaction problems (PCSP's), such as scheduling or sequencing, do not lend themselves easily to automated solving. A solution to a PCSP typically must satisfy a set of hard constraints to be valid or feasible, and must minimize the costs associated with violating one or more conflicting soft constraints (i.e., the solution must present the best trade-off among the conflicting soft constraints). The soft constraints cannot be all completely or simultaneously satisfied in a solution. Thus, each soft constraint is allowed to be violated at a cost (i.e., partially satisfied) and the best solution is a valid solution (i.e., satisfying all hard constraints) which minimizes the costs incurred with violating the soft constraints and the degree of violation thereof.

PCSP's, such as the scheduling (or sequencing) of products for manufacture present a particularly difficult type of problem—a computationally complex problem, referred to as an NP-complete problem (see Garey, M. and Johnson, D. , *Computers and intractability: a guide to the theory of NP-completeness*, Freeman, 1979.), for which search techniques that deterministically and exhaustively search the space of possibilities fail to generate a viable solution in a realistic time period. For the generic problem of assigning N tasks to M resources with a particular ordering of tasks at each resource, the number of possible solutions is

$$\binom{N+M-1}{M-1}N!$$

which implies super-exponential growth as a function of the number of tasks and resources. In addition to sheer size, this search space has a more complex topology than a Euclidean space or, when M >1, a space of permutations.

Exhaustive search (for the best sequence) is impractical for most of the real-world sequencing problems which often have a sequence length ranging from the 100's to the 1000's and are subject to many conflicting constraints. Traditional algorithmic techniques, such as dynamic programming (see Bellman, R. E., and Dreyfus, S. E., *Applied dynamic programming*, Princeton University Press, 1962), and branch-and-bounds (see Lawler, E. L., and Wood, D. E., "Branch and bounds methods: A survey," *Operations Research*, 14, 699–719, 1966) fail due to their lack of scalability. The heuristic methods developed to date are complicated by the details of a particular task, and the algorithmic consideration of the specific constraints is embodied in what amounts to a domain specific expert system. Such heuristic techniques are often highly specific to particular application domains and thus are not useful as general techniques for solving PCSP problems.

Real-world scheduling/sequencing problems present special difficulties not found in the less than real-world problems addressed in the art to date. These difficulties include: (1) the size and complexity of the real-world search space, which is far more formidable than the search typically defined in less than real-world problems; (2) the dynamic process inherent in real-world scheduling/sequencing problems (schedules remain valid only for a limited amount of time; after a certain duration, the world generally has changed enough that the scheduling algorithm has to find a different schedule; additionally, there are time constraints on how long it can take to schedule and reschedule that place limits on the amount of computation that can be performed by a scheduling algorithm; and (3) the different domains and applications present in real-world scheduling problems, which require solutions of different variations of the scheduling problem (these variations arise from a number of different sources, including: differences in the types of hard constraints, such as relative and absolute temporal restrictions, and resource capabilities constraints; the need for additional information beyond an ordered assignment of tasks to resources, such as absolute times, routes traveled, and manufacturing plans; and different sets of evaluation criteria, such as cumulative response time, throughput, time span, and cumulative employee satisfaction.) Real-world scheduling/sequencing algorithms thus must be flexible enough to accommodate different conditions and able to adapt to changes.

Evolutionary computation (abbreviated as "EC"), a general, stochastic framework inspired by the intrinsically robust search and optimization paradigms of biological evolution, presents the most promising direction in solving real-world PCSP's. EC algorithms have been applied to many hard optimization problems where classical methods (e.g., gradient search, linear programming, etc.) have failed to provide good solutions. See, e.g.. Back, T., Fogel, D. B., and Michalewicz, Z, eds., *Handbook of Evolutionary Computation*. New York: Oxford Univ. Press and Institute of Physics, 1997.

All EC algorithms share the same basic high-level philosophy and structure. In any EC algorithm, a "population" of individuals, called "chromosomes," is maintained. Each

chromosome represents a potential solution to the problem to be solved. An "initialization process" is incorporated to create the initial population (i.e., the first generation of chromosomes). A "fitness measure" representing certain optimization criteria is employed to evaluate how fit or optimized each chromosome is, and a "selection process" is used to select chromosomes for "reproduction" based on their fitness values. The reproduction process applies "genetic operators" to the selected chromosomes to create probabilistically perturbed variants, called "offspring," among which are likely fitter chromosomes. A unary genetic operator alters a (parent) chromosome in some fashion probabilistically, and is called a "mutation." A binary genetic operator combines probabilistically determined portions of two parent chromosomes to produce two offspring, and is called a "crossover." The offspring chromosomes are then evaluated by the fitness measure and, based on their values of fitness, are used to replace certain worse chromosomes in the population to form a new generation.

The general EC procedure can be outlined as follows, where t denotes generation index and P(t) denotes the population of chromosomes at t:

```
Begin
  t=0;
  Initialization (to create P(t) typically randomly);
  Repeat
    Fitness evaluation of P(t);
    Selection of chromosomes in P(t);
    Reproduction to produce a P(t+1);
    t=t+1;
  Until certain termination condition is satisfied (when
    t=T)
End.
```

By repeating the above evaluation-selection-reproduction loop, the population can be "evolved" to fitter populations (Darwinian survival of the fittest). Note that in the process, P(t) remains of the same size, i.e., has the same number of chromosomes. After a certain number of generations (which is either predetermined, or decided based on specified "termination" conditions), the best chromosome in the final population represents the optimal or near-optimal solution to the problem.

Such a EC paradigm is inherently parallel (as a whole population can evolve simultaneously) and robust (as it is a probabilistic or randomized process). It combines exploitation of the most promising search area, through fitness-governed selection and reproduction, and exploration of the broadest search space through randomness.

Such a EC paradigm is also general in the sense that only the fitness function used to evaluate how "good" a chromosome is requires problem-specific knowledge or information; the other components can be relatively problem-independent. For example, initialization is typically done by randomly generating chromosomes. A typical method of chromosome selection is "roulette wheel" sampling in which each chromosome is assigned a slice of a circular "roulette wheel," with the size of the slice being proportional to the chromosome's fitness. The wheel can then be spun N times, where N is the number of individuals in the population. On each spin, the individual under the wheel's marker is selected to be in the pool of parents for the next generation.

On the other hand, one can also design each component of the EC paradigm (as outlined above) differently to suit special needs of a particular problem. Thus, the EC paradigm is both general and flexible.

Indeed, different EC algorithms are characterized by different designs of the basic components: (1) representation of chromosomes, (2) selection strategy, (3) reproduction strategy (including genetic operators), and (4) problem-dependent criteria and method for fitness evaluation of chromosomes. For example, three pioneer classes of EC algorithms, Genetic Algorithms (GAs) (see Holland, J., *Adaptation in Natural and Artificial Systems*, University of Michigan Press, Ann Arbor, 1975), Evolutionary Programming (EP) (see Fogel, L., et al., *Artificial Intelligence through Simulated Evolution*, Wiley, New York, 1966), and Evolutionary Strategies (ESs), differ in these components. In GAs, which are perhaps the most influential form of EC algorithms, a chromosome has the structure of a bit string, and crossover is emphasized as a major operation, whereas in EP and ES, a chromosome represents a string of real numbers, and mutation is emphasized.

In early applications of EC, a concrete problem description was typically mapped into a standard, well-known chromosome representation, such as a bit string in a classical GA, and the problem was then solved by a standard, well-known EC algorithm, such as a classical GA. This, however, has many limitations. Although bit string (or binary) representation has facilitated the development of the theoretical foundation of GAs, bit string representation severely limits the range in which the algorithm can operate. It is thus ineffective in problems where the desired solution is hierarchical, or where the size and shape of the solution is unknown in advance. In a classical GA, there is one mutation operator and one crossover operator. The mutation operator flips a randomly selected bit in a chromosome to produce a new offspring. The crossover operator cuts the two chromosomes into two parts at the same (randomly selected) position and swaps the second parts of the two chromosomes to produce two offspring chromosomes. These operations, at the level of bits, usually make evolution too slow in converging to a satisfactory solution.

More recent approaches emphasize the use of "natural" chromosome structures which best describe the class of problems to be solved, and accordingly, the use of problem-sensitive genetic operators (see Michalewicz, Z. *Genetic algorithms+data structures=evolution programs*, 3rd ed., Springer-Verlag, 1996). An important question (and problem) inherent in these approaches, however. is to what extent should the problem-specific knowledge or information be embedded in the algorithm to achieve a good balance of efficiency and general applicability. Trying to answer this question well is a continuing struggle for EC practitioners.

Among the problems attacked by EC methods, scheduling problems have drawn considerable attention, see Special Issue: Evolutionary Algorithms for Scheduling, *Evolutionary Computation*, 6(1), 1998, including the specific problem of car sequencing, see Warwick, T. and Tsang, E. P. K., "Tackling car sequencing problems using a generic genetic algorithm," *Evolutionary computation*, 3(3): 267–298, 1995. However, the car sequencing problems considered to date constitute unrealistically simplified versions of the real-world problem, having no more than one hard constraint and one type of soft constraints. The approaches considered to date are also not suitable for real-world sequencing problems, which have many different kinds of hard and soft constraints, because with multiple constraints, they do not result in a feasible or valid solution within a finite time period. The prior art crossover operator also can result in severe violation of hard constraints in the offspring produced, and repairing the infeasible offspring can be very expensive.

In sum, there is no effective method, EC-based or not, in the prior art which is able to handle the complex real-world

sequencing problems successfully. The present invention overcomes the problems and insufficiencies in the art in a novel manner.

BRIEF SUMMARY OF THE INVENTION

In view of the shortcomings and inabilities of the existing methods for solving real-world sequencing problems, the present invention provides a general method of sequencing manufacturing tasks which employs an EC algorithm to determine near-optimal solutions to a partial constraint satisfaction problem ("PCSP"). A preferred embodiment comprises a method for determining within a finite time period near-optimal sequencing of manufacturing tasks subject to user-given hard and soft constraints.

It is to be understood that the foregoing general description and the following detailed description are exemplary and explanatory only, and not restrictive of the invention as claimed.

DESCRIPTION OF THE DRAWINGS

The present invention will be understood more fully from the detailed description given below and from the accompanying drawings which, however, should not be taken to limit the invention to the specific embodiment but are for explanation and understanding only.

FIG. 1 depicts a chromosome (S) representation of a product sequence: $S=\{S_1, S_2, \ldots, S_N\}$ be a sequence of N products of M types. Assume that an assembly schedule must be generated for a set of N vehicles.

FIG. 2 illustrates a mutation operation.

FIG. 3 illustrates a crossover operation.

FIG. 4 illustrates the preferred method of the invention.

FIG. 5 illustrates the basic operations of an automobile manufacturing sequence.

DETAILED DESCRIPTION OF THE INVENTION

The present invention teaches a method of solving partial constraint satisfaction problems, in particular a method of sequencing products for manufacture such that the sequence produced is a near-optimal or optimal sequence, meaning that direct cost associated with various labor, processes and parts inventory are minimized and equipment and floor space utilization are maximized. An example of sequencing products for manufacture is the manufacture of automobiles, in which vehicles are sequenced to go through a series of operations such as body forming, painting, component assembly (such as installing radios, seats, etc.) and final assembly (adding trim and chassis).

In the present invention, in order to be valid or feasible, a solution sequence must satisfy any hard constraints imposed. In addition, there are a number of desirable, but conflicting, constraints (referred to as "soft constraints") which cannot all be completely satisfied in a sequence at the same time. Thus, soft constraints are allowed to be violated at a cost, or partially satisfied. The optimal solution sequence will be a valid sequence (i.e., satisfying all hard constraints) which also minimizes the degree and costs of violating all soft constraints (i.e., providing the best trade-off among the conflicting soft constraints).

The present invention applies to sequencing problems which include some or all of the constraints described below. Constraints other than those described below may also be included.

The method of the present invention comprises: A computer implemented method for generating an optimized or nearly-optimized sequence of "N" number of products for manufacture, where said products are of "M" number of distinct types with a fixed number ("N_t") of each type being desired and each product type comprises a unique array ("Q") of distinct features, wherein said manufacture is optionally constrained by one or more of the following constraints: the production requirement for each product type, feature-based position equations, and feature-based position inequalities, wherein each of said constraints is individually designated as either a hard constraint which cannot be violated, or as a soft constraint which can be violated at a predetermined cost; said method comprising:

generating an initial population of chromosomes, wherein each chromosome represents a feasible sequence of products of various types for manufacture, feasibility depending on satisfaction of all of said hard constraints;

associating a fitness value with each chromosome, said fitness value being a function of the predetermined cost associated with the degree of violation of each of said soft constraints;

sorting said chromosomes based on the fitness value associated with each chromosome; and

applying iteratively to the population of chromosomes a reproductive process, comprising (1) selection of a genetic operator, (2) selection of one or two chromosomes, the number of chromosomes to be selected correlating with the selected genetic operator, (3) application of the selected genetic operator to the selected one or two chromosomes to cause generation of one or two offspring, (4) insertion of one offspring chromosome into the sorted population, and (5) discard of one of the least desirable chromosomes in the population;

said iterative process being continuously run until the fitness value for the best chromosome satisfies one or more known criteria or until a pre-determined time has elapsed.

The problem description—described above as "N number of products for manufacture, where said products are of "M" number of distinct types with a fixed number ("N_t") of each type being desired and each product type comprises a unique array ("Q") of distinct features" can be formulated as follows:

Let $S=\{S_1, S_2, \ldots, S_N\}$ be a sequence of N products of M types, where each type t is described by a tuple of Q features $(f_1, f_2, \ldots, f_Q)$, such that f_i (i=1, 2, . . . , Q) $\in D_i$, where D_i is the domain of feature values for feature i. S is subject to hard constraints, which may or may not include some or all of the following types of hard constraints:

A. Production requirements: for each product type t (=1, . . . , M), N(t) products have to be in the sequence S (i.e., have to be produced). Thus $N=\Sigma N(t)$. (Alternatively, the product requirement can be included as a soft constraint.)

B. Feature-based position equations: a set of equations which determine product positions based on values of a subset or the whole set of type features. This kind of constraints take into account the limitations present in the means of moving the sequence in the work space (e.g., the availability and locations of holders to carry products to various processes and the ways processes can be applied to the products).

C. Feature-based position inequalities: a set of inequalities which determine the ranges of product positions based on values of a subset or the whole set of type features. This

kind of constraint takes into account the time/timing requirements to operate on certain products or their features (e.g., the time frame when workers of certain specific expertise will be available or certain resources/equipment can be available).

In addition, S may (or may not) also be subject to some or all of the following types of soft constraints (i.e., conflicting constraints which cannot be satisfied simultaneously and completely):

A. Feature-based distance equations: a set of equations which determine product distances in the sequence based on values of a subset or the whole set of type features. This kind of constraint often characterizes the need to either group together or spread products with certain features in a sequence for efficient processing (e.g., products requiring the same part are not put too close in a sequence so that there will be sufficient time for the worker or process to install the part without causing a stop of the sequence at the installation point).

B. Feature-based distance inequalities: a set of inequalities which determine the ranges of product distances in the sequence based on values of a subset or the whole set of type features. This kind of constraint usually takes into account the fact that sometimes products do not have to go through a certain process strictly sequentially, rather the process may be applied to products out-of-order so long as the substituted product possess the same or similar features of the replaced product.

For a given feasible and valid sequence, a cost is computed to measure the violation of each soft constraint or each type of soft constraints in the sequence, and the total cost C of the entire sequence is a combination of such individual costs. C can be formulated as a weighted sum of individual costs, where each weight determines the contribution of the corresponding cost to the total cost C. The method of the present invention then finds the optimal or nearly optimal sequence, that being the best sequence which satisfies all hard constraints and minimizes the total cost C for violation of soft constraints.

The present invention utilizes an EC algorithm characterized by the following components:

Chromosome representation: A population of individual chromosomes, each chromosome being an ordered list representing a feasible and valid product sequence (see FIG. 1), wherein each element in the list corresponds to a product in the sequence. The length of a chromosome is N, the total number of products in a sequence.

Fitness evaluation: The fitness of a chromosome (i.e., a sequence) is measured by the total cost C (as discussed above) for violating soft constraints. Thus, the fitter a chromosome, the lower its total cost.

Initialization: The initial population of chromosomes must include some randomly generated valid sequences (i.e., only sequences which satisfy all hard constraints). The population can consist of only randomly generated valid sequences, or it can also include some pre-determined sequences based on problem-specific heuristic knowledge ("imported" sequences).

Genetic operators: Genetic operators are used to alter chromosomes in order to create new and potentially fitter chromosomes. Two operators are preferred: one mutation operator and one crossover operator. The mutation operator performs the following operation on a single chromosome (i.e., a single sequence): first, randomly select two positions where the products are not of the same type; if swapping the two products does not violate any hard constraint, the products are swapped; otherwise, either two new positions are selected and the process repeated, or alternatively the process is aborted (see FIG. 2). Note that operation of the mutation operator does not cause any violation of the hard constraint of production requirements.

The crossover operator operates on two chromosomes to generate two new chromosomes (called offspring). It starts by randomly selecting a sequence position and checks if the corresponding products in the two sequences are of the same type. If they are not, the corresponding products are swapped to produce two offspring. Note that as the swapping occurs at the same position, the feature-based position and position range constraints are not violated. Since the swapping inevitably violates the hard constraint of production requirements in the two offspring by decreasing the number of one type of product by 1 and by increasing the number of the other type of product by 1 at the same time, another swapping is needed to repair the offspring. This second swapping (referred to as a "Repair Operation") is likely to happen between products at different positions, as shown in FIG. 3, but again, it will not violate the feature-based position or position range constraints as in the first swapping.

Reproduction: Reproduction refers to the process of generating new chromosomes. In the present invention, this process consists of first selecting a genetic operator, and then selecting either one (if the mutation operator is selected) or two chromosomes (if the crossover operator is selected) for reproduction. Each operator has a probability, and a roulette wheel is constructed based on the operator probabilities. The probability value of an operator can either be pre-determined based on problem-specific knowledge, or can be adjusted during the evolution process based on the operator's usefulness (i.e., performance) in past generations. An operator is selected by "spinning" the wheel.

To facilitate the selection of chromosomes, the initial population of chromosomes are sorted in the ascending order of their fitness (i.e., cost) values. A roulette wheel is built with the size of the slice for each chromosome proportional to its rank in the sorted list. Then each time a selection is done by "spinning" the wheel a chromosome is selected with a probability proportional to its rank. After the genetic operation, the newly generated offspring (or one of the two offspring generated from crossover with repair) is evaluated (i.e., its fitness value is calculated) and is inserted into the sorted population based on its fitness value; one of the worst chromosome in the population is then discarded to maintain the original population size. Such a population now represents a new generation and remains sorted.

Overall algorithm: The algorithm starts with an initialization process in which the population is generated. The population is then evaluated as sorted as described above. The reproduction process described above is then repeated until either a set of termination conditions are satisfied, or a pre-determined amount of time has elapsed. The algorithm flow chart is as shown in FIG. 4.

Detailed Description of an Illustrative Embodiment

The nature and variants of the present invention are illustrated by the following example. Although the example is limited to one industry specific embodiment of the invention, the invention itself should not be construed as being so limited. Additionally, (and it should be apparent) various variations and modifications may be made to the invention, with the attainment of some or all of the advantages of the invention. It is the subject of the appended claims to cover these and such other variations and modifications as come within the true spirit and scope of the invention.

The Vehicle Sequencing Problem

The vehicle sequencing problem represents a special instance of the general sequencing problem described above. The primary objective of vehicle sequencing is to develop a vehicle assembly sequence which is preserved throughout the body shop, paint, and final assembly processes, is unlikely to be disrupted, and which simultaneous minimizes the direct costs associated with labor, parts, inventories, and paint costs, while increasing equipment and floor space utilization.

Recently, car sequencing has taken on greater significance as the complexity of vehicle assembly has increased due to automotive manufacturers' desire to accommodate consumers' individual preferences vis-a-vis car exterior colors and features, such as sun roofs, car interior colors, fabrics and options.

The Nature of Vehicle Sequencing

Currently, automobile manufacturing companies offer customers a wide variety of vehicle configurations within each model. The possible combinations of body styles, powertrains, interior and exterior colors, and free-standing options can number in the tens to hundreds of thousands of unique vehicles. This presents a very complex scheduling problem when determining the ideal sequence of vehicles to be built on a given day. However, it is an important problem to solve, since an optimum sequence will reduce direct labor, parts inventories and paint costs, while increasing equipment and floor space utilization.

In the past, off-line repairs of defects in the upstream body shop created an unpredictable sequence of vehicles for the paint shop, while off-line paint defect repairs resulted in an even less predictable bank of vehicles for the final assembly (trim and chassis) line sequence. Poor sequences entering the paint bank cause a significant waste of materials. For instance, each time the color of a vehicle changes from its predecessor in the enamel paint booth, the previous paint color must be purged from the paint system gun, wasting paint and cleaning solvent. Poor sequences can also result in the occurrence of unnecessary costs in the final assembly line. The cost of poor sequences with respect to high labor value options is direct labor. For example, consider an assembly line which produces sixty vehicles per hour and contains a moonroof installation station. For a randomly arranged sequence of vehicles, if each installation requires 1.5 minutes of direct labor by two workers, a second worker is needed in the moonroof station to stay within the 1 minute of cycle time. In a "good" assembly sequence, at least one non-moonroof vehicle will be spaced between any two moonroof vehicles, such that one worker will be able to keep up with the workload and return to the beginning of his station after each installation. Good sequences also require less floor space per individual station because workers are not forced as far out of their station by the needs of consecutive high labor value jobs. The sequencing efficiencies are lost when the optimal manufacturing sequence is disrupted by missing vehicles at the point where units are also moved from the paint bank to the start of the final assembly line. Finally, an unpredictable sequence requires the inventory of parts to account for all possible vehicle configurations.

These types of inefficiencies can be avoided using the method of the present invention. For instance, with the optimized sequencing solution of the present invention in place, suppliers can ship the exact parts at the exact time needed and significantly reduce inventory holding costs. Additionally, since units stored in the paint bank have no powertrain, trim or options, and are distinguishable only by their body-in-white (BIW) characteristics and paint color, the present invention reduces inefficiencies by providing for substitution between any missing unit and another unit in the paint bank with the same BIW and color but a later sequence number. The missing unit then gains additional time to arrive in the paint bank before it causes a break in the vehicle sequence.

The combination of BIW (e.g.,. sedan/wagon, two or four door, moonroof, and seating configuration) and paint color (e.g., body color and any two-tone accent color or paint stripe) is called a painted body. Depending on the vehicle line there can be up to a thousand unique painted bodies. To facilitate substitution, the method of the present invention provides for a "back-up" for each vehicle consisting of an identical painted body in the paint bank at the time that it must be moved to the final assembly line. To do so, requires spacing identical painted bodies at a fixed interval which relates to the paint bank size. [Substitution has cost benefits resulting from smaller paint banks and less inventory in the paint banks, as well as optimized in-line vehicle sequencing.] Painted body spacing, however, adds a great deal of complexity to the sequencing problem, since the effect is to add one constraint for each unique painted body. For a typical plant, this will increase the number of constraints from less than 20 to over 100. The present invention, however, provides a feasible method for sequencing of this high level of complexity.

Representation

The product here is a vehicle, and the type of the vehicle is determined by the following features: body type, paint type/color, and features characterizing option types (such as radio, car phone, seat cover, etc.). A customized vehicle type is then described by a tuple of such features, and each set of feature values describes a unique vehicle type. Thus, the problem may be represented as follows:

Given N cars of M types, which require a total of K options, as specified by

$$O(j,k)=\begin{cases} g & \text{if car type } j \text{ requires option } k \text{ of grade } g \\ 0 & \text{otherwise} \end{cases}$$

where j=1, . . . , M and k=1, . . . , K, find a schedule S as a sequence of the N cars, satisfying the hard constraints: (1) production requirements; (2) fixture constraints; and (3) rotation range constraints; while optimizing the following: (1) optional spacing; (2) painted-body spacing; and (3) paint blocking.

Each vehicle type j,j=1, . . . , M, is determined by a tuple of attributes (bj, pj, pbj, Oj), where

bj is the index of the car body type,

pj is the index of the paint type,

pbj is the index of the painted-body type, and

Oj is a set of option-grade pairs as defined by the function O (j,k).

Production requirements, fixture constraints, and rotation range constraints are three types of hard constraints often considered in a real-world vehicle sequencing problem. The first, production requirements, is as described in the general problem formulation earlier, and can be formulated in the problem as Prj cars for car type j, where j=1, . . . , M. Note that

$$N = \sum_{j=1}^{M} Pr_j.$$

The second constraint, which is referred to as the fixture constraint, is a special case of the general feature-based position equations constraint. The fixture constraint requires that a vehicle of certain body type (e.g., sedan or wagon, or different car model) can only be placed at some specific positions in the sequence. The fixture constraint can be formulated in the problem as a fixed body type sequence x_1 $x_2 \ldots x_N$, where $x_i \in \{1, \ldots, B\}$ for i=1, . . . , N, is required, where B is the total number of body types.

The third constraint, referred to as the rotation range constraint is a special case of the general feature-based position inequalities constraint. The rotation range constraint requires that vehicles with certain feature values must be produced within a certain time frame, which can be translated as that they must be placed within a certain range of positions in the sequence. The rotation range constraint can be formulated in the problem as:: $i_{j1}<r(i)<i_{j2}$ where r(j) is the position of the car type j, and $1<i_{j1}<i_{j2}<N$.

The hard constraints are consistent with one another and can all be satisfied at the same time.

The three types of soft constraints, optional spacing, painted-body spacing, and paint blocking, are generally included in real-world vehicle sequencing problems. The option spacing constraint is a special case of the general feature-based distance inequalities constraint. It specifies, in this instance, that any two vehicles sharing certain feature values should be separated by a minimum distance, i.e., the smallest distance d between such two vehicles is greater than a d_{min} (i.e., $d>d_{min}$). For example, if two vehicles require the same type of radio and no other vehicle between them in the sequence requires that type of radio, the distance between them should allow sufficient time for installation of the radio on one vehicle before the second vehicle arrives for installation of the same radio.

Optional spacing can be formulated in the problem in the following manner: if there are N_k vehicles requiring option k in the sequence of N vehicles, the ideal spacing for option k is N/N_k. Less spacing incurs a cost. Cost for option k, thus, would be:

$$C_{os}(k) = \sum_{i=1}^{\#violations} \left(\frac{ideal_k - actual_i + 1}{ideal_k} \right)^3$$

and total cost for optional spacing is a weighted sum of costs for individual options:

$$C_o = \sum_{k=1}^{K} a_k C_{os}(k)$$

where a_k is the priority/weight for option k and K is the total number of options.

The painted-body spacing constraint is also a special case of the general feature-based distance inequalities constraint. It specifies both a minimum and a maximum distance between two vehicles of the same painted-body type, i.e., the combination of a basic body type (without powertrain, trim, or options) and the paint color. For each painted-body type, such a constraint is in the form of $d_{min}<d<d_{max}$, where d is the smallest distance between two vehicles of that painted body. The painted-body spacing constraint ensures that there is always at least one substitute in the paint bank for any painted body accidentally missing from the sequence at the scheduled time of its painting. This substitution allows for efficiency to be gained by preventing the break of sequence due to the missing unit and also allows for additional time for the missing unit to arrive at the paint bank. The determination of the distance bounds (d_{min} and d_{max}) depends on the size of the paint bank.

Painted-body spacing can be formulated in the problem as: $n \in [L,U]$ vehicles of identical painted bodies should be in the paint bank of size A at any time. Violation incurs a cost. N_p (i,j) is the number of vehicles of painted-body j in the A vehicles starting from position i.

The cost associated with j at i is, thus:

$$C_{pl}(i, j) = \begin{cases} L - N_p(i, j) & \text{if } N_p(i, j) < L \\ N_p(i, j) - U & \text{if } N_p(i, j) > U \\ 0 & \text{else} \end{cases}$$

and total cost for painted-body spacing:

$$C_p = \sum_{i=1}^{N} h_j C_{pl}(i, j)$$

where h_j is the weight for painted-body type j and contains painted-body type information for position i.

The paint-blocking constraint is a special case of the general feature-based distance equation constraint. It specifies that vehicles of the same exterior paint color be grouped together, i.e., d=1, where d is the smallest distance between two identical painted bodies. This constraint minimizes the financial cost associated with paint gun purge every time paint colors are switched. This constraint can be formulated in the problem as: V vehicles of the same paint type are preferred to form a consecutive subsequence (block). Violation incurs a cost. If N_m is the length of the m-th same-paint subsequence in the vehicle sequence, then the associated cost is:

$$C_b(m) = \begin{cases} 1 & \text{if } N_m \% V > 0 \\ 0 & \text{if } N_m \% V = 0 \end{cases}$$

where N_m % V returns the remainder when N_m is divided by V, and the total cost for paint blocking is a weighted sum of costs associated with individual same-paint subsequences:

$$Cpb = \sum_{m=1}^{D} w_m C_b(m)$$

where D is the total number of paint changes.

Clearly the soft constraints described above conflict with one another, and it is difficult or even impossible to satisfy any one of them completely. Accordingly, violation of a soft constraint is allowed at a pre-determined cost.

Given a vehicle sequence, as in the general formulation of the sequencing problem described earlier, the total cost associated with the soft constraints is a weighted sum of individual costs, where each weight determines the contribution of the corresponding individual cost of the total cost. The fitness of each chromosome is thus a function of the cost function measuring the degree of violation of soft constraints: $Cost=c_0 C_o+c_1 C_p+C_2 C_{pb}$. The coefficients c_i, i=0, 1, 2 can characterize the priorities of the three kinds of costs.

The problem can now be solved with the algorithm of the present invention to find the best sequence of vehicles which satisfies all hard constraints and minimizes the total cost for violation of soft constraints.

The algorithm of the present invention was tested on sample data sets provided to the inventor by a major United States automobile manufacturing company. The results of the test runs are presented in the Examples below.

Example One (Data Set 1)

Data Set 1: 512 vehicles of 100 types, 1 body type, 12 paint types, 28 painted body types, and 9 options.

TABLE 1

System Parameters

Population Size	Crossover Probability	Mutation Probability
5	0.6	0.4

Results for Data Set 1

TABLE 2

Total time

	generation = 10,000
Cost With Optional Spacing Only	about 2.5 min.
All costs	about 18 min.

TABLE 3

Costs with optional spacing only

Initial Population	Best Initial Result	Best Final Result (generation = 20,000)
Randomly Generated	333.068	23.684
Included Sample*	25.080**	8.967

*a pre-determined "good" sample sequence is included in the otherwise randomly generated initial population.
**is also the cost of the sample sequence.

TABLE 4

Costs (randomly generated initial population)

Costs (weights)	Best Initial Result	Best Final Result (generation = 20,000)
Optional Spacing (1)	353.52	46.45
Painted Body Spacing (0.1)	8864	7755
Paint Block (1)	450	294

Example Two (Data Set 2)

The system parameters for this run were as set forth in Table 1 in Example One.

Data Set 2: 852 vehicles of 167 types, 2 body types, 13 paint types, 1 painted body type, and 20 options.

Results for Data Set 2

Total time: about 5 minutes for 10,000 generations.

TABLE 5

Costs (with optional spacing only)

Initial Population	Best Initial Result	Best Final Result (generation = 20,000)
Randomly Generated	5.6×10^{16}	5.67×10^{15}
Included Sample(*)	6.45×10^{15}(**)	6.11×10^{15}

(*)a pre-determined "good" sample sequence is included in the otherwise randomly generated initial population.
(**)is also the cost of the sample sequence.

TABLE 6

Costs (with paint blocking only)

Initial Population	Best Initial Result	Best Final Result (generation = 10,000)
Randomly Generated	741	482
Included Sample	471*	385

*is also the cost of the sample sequence.

The algorithm of the present invention was also implemented and used in one real-world vehicle manufacturing plant with excellent results: the number of units in the paint bank without an identical painted body back-up being reduced by an average of 36%, while the rest of the scheduling objectives were also achieved.

It will be appreciated that many modifications may be made to the system described above without departing from the scope of the present invention. For example, possible modifications include:

A. Modifications to the way the values of the system parameters are assigned, such as the population size, the operator probabilities, the coefficients in the fitness measure, etc. While such values may be determined in an ad hoc way based on experience, the values assigned to system parameters can also be adapted based on their influence to the system and the system performance, which indirectly reflects the characteristics of the specific problem being attacked. Such a self-adapting system allows use of the system without requiring the users to be experts with insights of system and even the problem. One example of a self-adapting version of the present invention is the use of weighted sum of costs related to known goals. Different assignments of these weights would significantly affect the behavior of the optimization system.

B. Use of additional genetic operations. Existing genetic operations can be incorporated into the present invention, for instance, heuristic strategies for initialization (note that the preferred embodiment uses random initialization), and dynamic termination conditions.

Having described the exemplary embodiments of the invention, additional advantages and modifications will readily occur to those skilled in the art from consideration of the specification and practice of the invention disclosed herein. Therefore, the specification and examples should be considered exemplary only, with the true scope and spirit of the invention being indicated by the following claims.

What is claimed:

1. A computer implemented method for generating an optimized sequence of "N" number of items, where said items are of "M" number of types with a fixed number ("N,") of each type being desired and each item type comprising an array ("Q") of features, wherein said sequence of items is

optionally constrained by one or more of the constraints, wherein each of said one or more constraints is individually designated as either a hard constraint which cannot be violated, or as a soft constraint which can be violated at a predetermined cost; said method comprising:

generating an initial population of chromosomes, wherein each chromosome represents a feasible sequence of items, feasibility depending on satisfaction of all of said hard constraints;

associating a fitness value with each chromosome;

sorting said chromosomes based on the fitness value associated with each chromosome; and

applying iteratively to the population of chromosomes a reproductive process, said reproductive process comprising application of a selected genetic operator to a selected one or more chromosomes to cause generation of one or two offspring said reproductive process being continuously run until the fitness value for the best chromosome satisfies one or more known criteria, wherein one or more genetic operators available for selection is assigned a probability of selection value and a roulette wheel scheme and a roulette wheel scheme utilizing these probability of selection values is used to choose operators.

2. The method of claim 1, wherein said items comprise products for manufacture.

3. The method of claim 1, wherein said types are distinct types.

4. The method of claim 1, wherein said features are distinct features.

5. The method of claim 1, wherein said constraints comprise one or more constraints selected from the group consisting of: a production requirement for each item type; feature-based position equations; and feature-based position inequalities.

6. The method of claim 1, wherein said fitness value is a function of the predetermined cost associated with the degree of violation of each of said soft constraints.

7. The method of claim 1, wherein said reproductive process further comprises: (1) selection of said genetic operator, (2) selection of one or two chromosomes, the number of chromosomes to be selected correlating with the selected genetic operator, (3) application of the selected genetic operator to the selected one or two chromosomes to cause generation of one or two offspring, (4) insertion of one offspring chromosome into the sorted population, and (5) discard of one of the least desirable chromosomes in the population.

8. A computer implemented method for generating an optimized sequence of "N" number of products for manufacture, where said products are of "M" number of distinct types with a fixed number ("Nt") of each type being desired and each product type comprising an array ("Q") of distinct features, wherein said manufacture is optionally constrained by one or more of the following constraints: the production requirement for each product type, feature-based position equations, and feature-based position inequalities, wherein each of said constraints is individually designated as either a hard constraint which cannot be violated, or as a soft constraint which can be violated at a predetermined cost; said method comprising:

generating an initial population of chromosomes, wherein each chromosome represents a feasible sequence of products of various types for manufacture, feasibility depending on satisfaction of all of said hard constraints;

associating a fitness value with each chromosome;

sorting said chromosomes based on the fitness value associated with each chromosome; and

applying iteratively to the population of chromosomes a reproductive process, said iterative process being continuously run until the fitness value for the best chromosome satisfies one or more known criteria or until a pre-determined time has elapsed, wherein said one or two chromosomes are selected using a roulette wheel, wherein said roulette wheel comprises a probability distribution derived from the relative ranking of the fitness values associated with each chromosome such that the probability for an individual chromosome to be selected is proportional to its relative ranking.

9. The method as set forth in claim 8 wherein said genetic operator comprises one of the of the following: a mutation operator and a crossover operator.

10. The method as set forth in claim 9 wherein one chromosome is selected if the genetic operator is a mutation operator.

11. The method as set for in claim 9 wherein two chromosomes are selected if the genetic operator is a crossover operator.

12. The method as set forth in claim 8 wherein said genetic operator is a crossover operator.

13. The method as set forth in claim 12 wherein application of said crossover operator comprises the steps of

selecting two parent chromosomes from the population,

randomly selecting a position index, said position index representing the location of a single product

comparing between the two parent chromosomes the single product located at said position index, and

if said two single products located, one in each parent chromosome at said position index, are not identical to one another, generating two offspring chromosomes identical to two parent chromosomes with the exception that the two single products located at the position index are swapped;

locating within each of the two offspring chromosomes the first sequential product that is identical to the swapped-in product; and

swapping between the two offspring chromosomes said sequential products.

14. The method as set forth in claim 8 wherein said genetic operator is a mutation operator.

15. The method as set forth in claim 14 wherein application of said mutation operator comprises the steps of randomly selecting two position indices, each position index representing the location of a single product, comparing the single product located at each position index, and if said single products are not identical, interchanging the single products so long as the resulting sequence satisfies all hard constraints.

16. The method as set forth in claim 8 wherein application of the selected genetic operator comprises the steps of randomly selecting two position indices, each position index representing the location of a single product, comparing the single product located at each position index, and if said single products are not identical, interchanging the single products so long as the resulting sequence satisfies all hard constraints.

17. The method as set forth in claim 8 wherein application of the selected genetic operator comprises the steps of

selecting two parent chromosomes from the population,

randomly selecting a position index, said position index representing the location of a single product

comparing between the two parent chromosomes the single product located at said position index, and

if said two single products located, one in each parent chromosome at said position index, are not identical to one another, generating two offspring chromosomes identical to two parent chromosomes with the exception that the two single products located at the position index are swapped;

locating within each of the two offspring chromosomes the first sequential product that is identical to the swapped-in product; and

swapping between the two offspring chromosomes said sequential products.

18. The method of claim **8** in which each of said one or more genetic operators available for selection is assigned a probability of selection value and a roulette wheel scheme utilizing these probability of selection values is used to choose operators.

19. The method of claim **18** wherein the probability of selection values are assigned in such a way that all operators are equally likely to be selected.

20. The method of claim **18** wherein the probability of selection values are assigned in such a way that the crossover operator is less likely to be selected.

21. The method of claim **18** wherein the probability of selection values are assigned in such a way that the crossover operator is more likely to be selected.

22. The method of claim **18** in which the probability of selection value assigned to each operator varies from iteration to iteration relative to the usefulness of each operator in previous iterations.

23. The method as set forth in claim **8** in which the initial population of chromosomes comprises randomly generated chromosomes.

24. The method as set forth in claim **8** in which the initial population of chromosomes comprises imported chromosomes.

25. A computer implemented method for generating an optimized sequence of "N" number of items, where said items are of "M" number of types with a fixed number ("N_i") of each type being desired and each item type comprising an array ("Q") of features, wherein said sequence of items is optionally constrained by one or more of the constraints, wherein each of said one or more constraints is individually designated as either a hard constraint which cannot be violated, or as a soft constraint which can be violated at a predetermined cost; said method comprising:

generating an initial population of chromosomes, wherein each chromosome represents a feasible sequence of items, feasibility depending on satisfaction of all of said hard constraints;

associating a fitness value with each chromosome;

sorting said chromosomes based on the fitness value associated with each chromosome; and

applying iteratively to the population of chromosomes a reproductive process, said reproductive process comprising application of a selected genetic operator to a selected one or more chromosomes to cause generation of one or two offspring said reproductive process being continuously run until the fitness value for the best chromosome satisfies one or more known criteria, wherein said genetic operator comprises a mutation operator, comprising the steps of randomly selecting two position indices, each position index representing the location of a single product, comparing the single product located at each position index, and if said single products are not identical, interchanging the single products so long as the resulting sequence satisfies all hard constraints.

26. A computer implemented method for generating an optimized sequence of "N" number of items, where said items are of "M" number of types with a fixed number ("N_i") of each type being desired and each item type comprising an array ("Q") of features, wherein said sequence of items is optionally constrained by one or more of the constraints, wherein each of said one or more constraints is individually designated as either a hard constraint which cannot be violated, or as a soft constraint which can be violated at a predetermined cost; said method comprising:

generating an initial population of chromosomes, wherein each chromosome represents a feasible sequence of items, feasibility depending on satisfaction of all of said hard constraints;

associating a fitness value with each chromosome;

sorting said chromosomes based on the fitness value associated with each chromosome; and

applying iteratively to the population of chromosomes a reproductive process, said reproductive process comprising application of a selected genetic operator to a selected one or more chromosomes to cause generation of one or two offspring said reproductive process being continuously run until the fitness value for the best chromosome satisfies one or more known criteria, wherein said genetic operator comprises a crossover operator, comprising the steps of:

selecting two parent chromosomes from the population,

randomly selecting a position index, said position index representing the location of a single product

comparing between the two parent chromosomes the single product located at said position index, and

if said two single products located, one in each parent chromosome at said position index, are not identical to one another, generating two offspring chromosomes identical to two parent chromosomes with the exception that the two single products located at the position index are swapped;

locating within each of the two offspring chromosomes the first sequential product that is identical to the swapped-in product; and

swapping between the two offspring chromosomes said sequential products.

* * * * *

US006636969B1

(12) **United States Patent**
Jakobsson et al.

(10) **Patent No.: US 6,636,969 B1**
(45) **Date of Patent: Oct. 21, 2003**

(54) **DIGITAL SIGNATURES HAVING REVOKABLE ANONYMITY AND IMPROVED TRACEABILITY**

(75) Inventors: **Bjorn M. Jakobsson**, Hoboken, NJ (US); **Joy C. Mueller**, Mainz (DE)

(73) Assignee: **Lucent Technologies Inc.**, Murray Hill, NJ (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

(21) Appl. No.: **09/299,327**

(22) Filed: **Apr. 26, 1999**

(51) **Int. Cl.**[7] **G06F 17/60**; H04L 9/30

(52) **U.S. Cl.** **713/180**; 713/160; 713/161; 713/181; 705/69; 705/75; 705/77; 705/78

(58) **Field of Search** 713/160, 161, 713/180, 181; 705/69, 75, 77, 78

(56) **References Cited**

U.S. PATENT DOCUMENTS

4,759,064	A	7/1988	Chaum	380/30
5,347,581	A	9/1994	Naccache et al.	380/30
5,787,172	A	7/1998	Arnold	380/21
5,796,833	A	8/1998	Chen et al.	380/25
5,889,862	A *	3/1999	Ohta et al.	705/69
5,901,229	A *	5/1999	Fujisaki et al.	380/30

OTHER PUBLICATIONS

A. Juels et al., "Security of Blind Digital Signatures," 1997, Advances in Cryptology, Crypto '97, pp. 150–164.*

D. Chaum and H. van Antwerpen, "Undeniable Signatures", Advances in Cryptology—CRYPTO '89; Santa Barbara, CA; Aug. 1989; Springer–Verlag, pp. 212–216.

M. Jakobsson and M. Yung, "Applying Anti–Trust Policies to Increase Trust in a Versatile E–Money System,", Advances in Cryptography—Proceedings of Financial Cryptology '97, pp. 217–238.

M. Jakobsson and M. Yung, "Distributed Magic Ink Signatures," Advances in Cryptology—Proceedings of Eurocrypt '97, pp. 450–464.

M. Jakobsson,"A Practical Mix," Advances in Cryptology—Proceedings of Eurocrypt '98, pp. 448–461.

M. Bellare, A. Desai, D. Pointcheval and P. Rogaway, "Relations Among Notices of Security for Public–Key Encryption Schemes," Advances in Cryptology—Proceeding of Crypto '98, pp. 26–45.

M. Jakobsson and M. Yung, "Revokable and Versatile Electronic Money," 3rd ACM Conference on Computer and Communications Security, 1996, pp. 76–87.

B. M. Jakobsson, "Privacy vs. Authenticity," A dissertation submitted in partial satisfaction of the requirements for the degree Doctor of Philosophy in Computer Science, 1997,pp. 1–54.

R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin, "Robust Threshold DSS Signatures," Advances in Cryptology—Proceeding of Eurocrypt '96, pp. 354–371.

* cited by examiner

Primary Examiner—Justin T. Darrow

(57) **ABSTRACT**

A method of authenticating electronic data is disclosed. In the preferred embodiment, when a Receiver makes a request to a Signer (e.g., a bank customer asks the bank to issue E-Coin), the Receiver includes a "hint generation value" in the request, and from the hint generation value, a "hint value" is derived and recorded on a signing transcript. The hint generation value is essentially an encrypted version of the request submitted from the Receiver to the Signer. When a merchant/Verifier transmits deposit signatures corresponding to spent E-Coin to be deposited, the transmitted signature is decrypted and blinded by the Signer in the same manner as that used to create the hint value. Thus, the encrypted incoming deposit signature from the merchant/Verifier should match the hint value stored on the signing transcript, confirming that the E-Coin is valid without revealing any identifying information about who spent the E-Coin, i.e., anonymity is preserved. If the incoming encrypted deposit signature does not match a hint value in the signing transcript, the bank immediately knows that counterfeit E-Coin is being circulated and can take the steps necessary to stop any further illicit transactions and attempt to identify the source of the corruption.

13 Claims, 2 Drawing Sheets

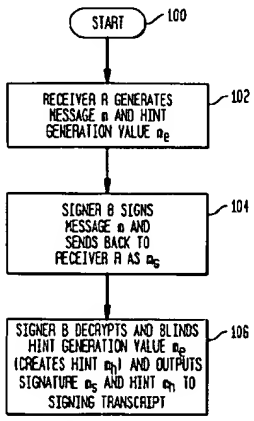


FIG. 1

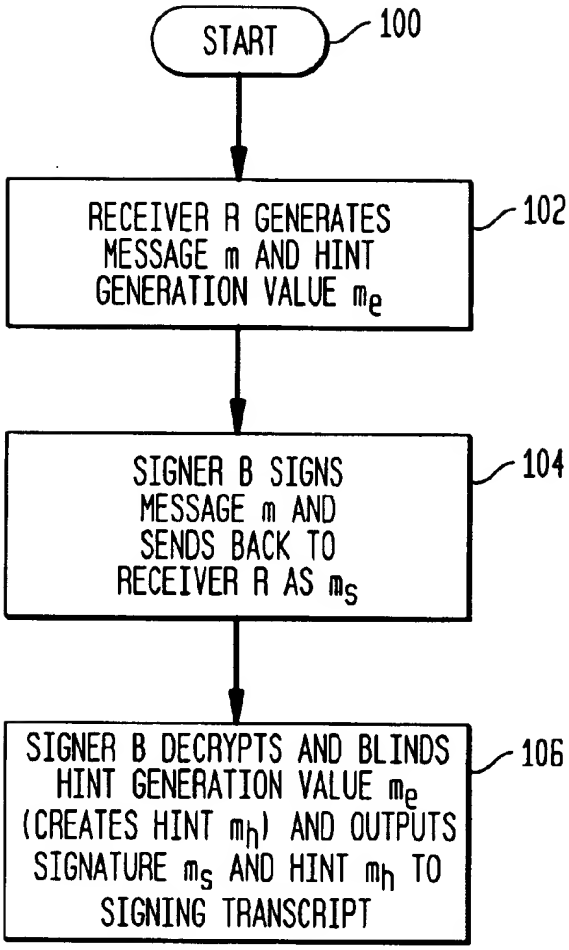
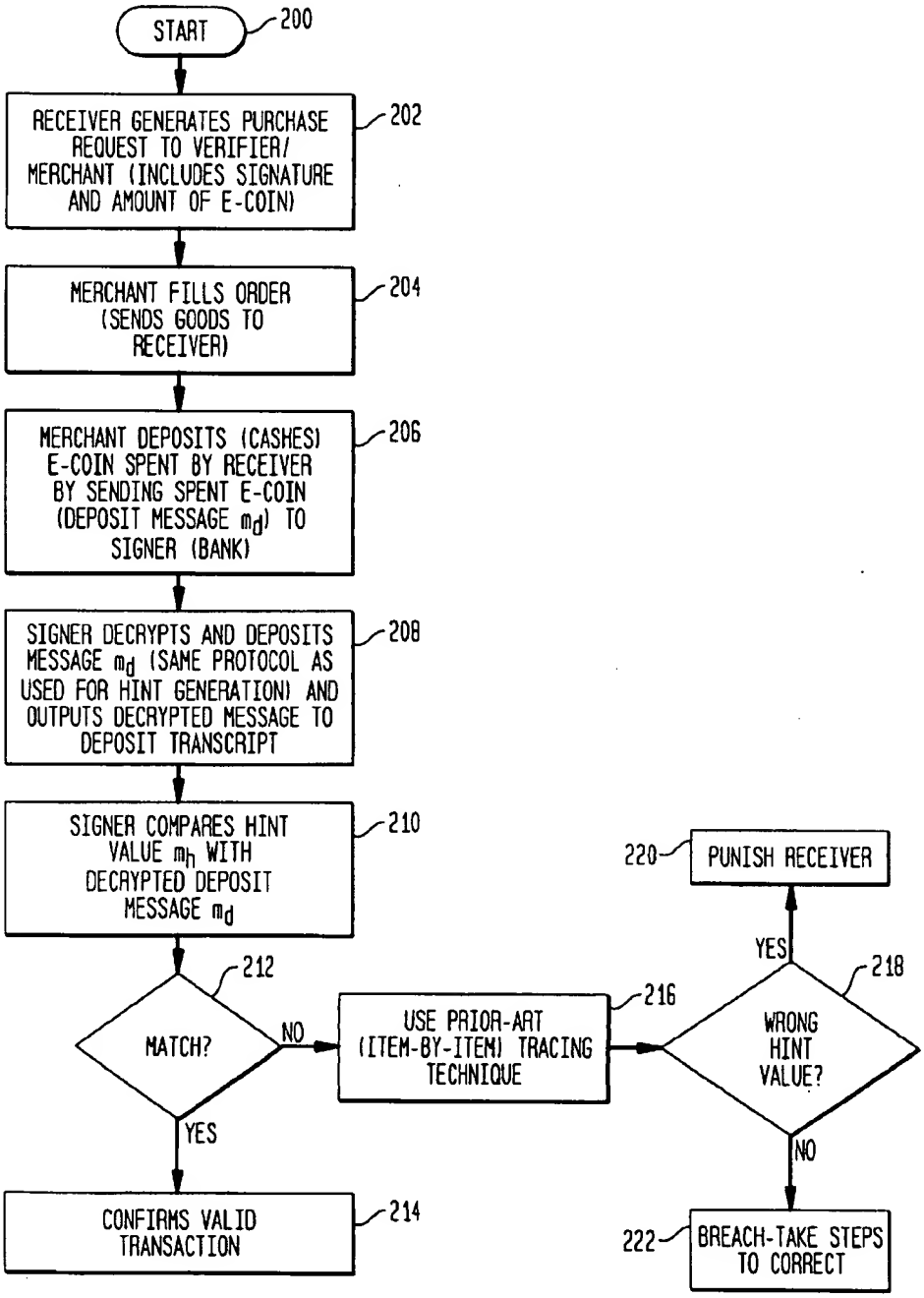

START
100
RECEIVER R GENERATES MESSAGE m AND HINT GENERATION VALUE m_e
102
SIGNER B SIGNS MESSAGE m AND SENDS BACK TO RECEIVER R AS m_s
104
SIGNER B DECRYPTS AND BLINDS HINT GENERATION VALUE m_e (CREATES HINT m_h) AND OUTPUTS SIGNATURE m_s AND HINT m_h TO SIGNING TRANSCRIPT
106

FIG. 2


START
200
RECEIVER GENERATES PURCHASE REQUEST TO VERIFIER/ MERCHANT (INCLUDES SIGNATURE AND AMOUNT OF E-COIN)
202
MERCHANT FILLS ORDER (SENDS GOODS TO RECEIVER)
204
MERCHANT DEPOSITS (CASHES) E-COIN SPENT BY RECEIVER BY SENDING SPENT E-COIN (DEPOSIT MESSAGE m_d) TO SIGNER (BANK)
206
SIGNER DECRYPTS AND DEPOSITS MESSAGE m_d (SAME PROTOCOL AS USED FOR HINT GENERATION) AND OUTPUTS DECRYPTED MESSAGE TO DEPOSIT TRANSCRIPT
208
SIGNER COMPARES HINT VALUE m_h WITH DECRYPTED DEPOSIT MESSAGE m_d
210
MATCH?
212
NO
USE PRIOR-ART (ITEM-BY-ITEM) TRACING TECHNIQUE
216
WRONG HINT VALUE?
218
YES
PUNISH RECEIVER
220
NO
BREACH-TAKE STEPS TO CORRECT
222
YES
CONFIRMS VALID TRANSACTION
214

DIGITAL SIGNATURES HAVING REVOKABLE ANONYMITY AND IMPROVED TRACEABILITY

FIELD OF THE INVENTION

This invention relates to the field of authentication of electronic data, and more specifically, to producing digital signatures that allow for revokable anonymity and the ability to detect the unauthorized use of a secret key without compromising anonymity requirements.

DESCRIPTION OF THE RELATED ART

Electronic commerce (E-Commerce) is one of the fastest growing segments of the Internet. All aspects of monetary transactions are being carried out electronically including banking, investing, purchase and sales, and the like. While the benefits of E-Commerce are many, certain precautions must be taken to prevent abuse and to ensure that the privacy of the participants is not compromised. Accordingly, authentication techniques utilizing "electronic signatures" and "secret keys" have been developed so that assurances can be made that the transactions requested are legitimate transactions. Many examples of such techniques can be found in the prior art (see, for example, M. Jakobsson and M. Yung, "Applying Anti-Trust Policies to Increase Trust in a Versatile E-Money System," Advances in Cryptology-Proceedings of Financial Cryptography '97, pp. 217–238.

In view of the need for protection of privacy, much of the research in the field of E-commerce has been focused on developing payment or signature schemes with revokable anonymity. Such schemes facilitate general anonymity with respect to transactions, but allow details of a particular transaction or user to be identified under appropriate circumstances (e.g., pursuant to a court order). For example, a set of "trustees" might possess the ability to remove the anonymity of a given user or transaction when all agree that there is reason to believe that the user is committing a crime or that a particular transaction is fraudulent.

"Blinding" is a technique utilizing electronic signatures by which the provider of a message for signing, e.g., a bank customer, can transform the message to be signed into a form which obscures the content of the message. Thus, the signer, e.g., a bank, can sign the transformed message and return it to the provider of the message, and the provider can transform the message in such a way that the result retains the digital signature property related to the original message content, but the result is not readily associated with the transformed message received by the signer. One example of such a technique is disclosed in U.S. Pat. No. 4,759,063 to Chaum, incorporated herein by reference.

To understand the known methods of revoking anonymity it is necessary to understand the mechanics of electronic transactions. For example, a typical electronic transaction will involve three participants: a Signer, typically a bank; a Receiver, typically a consumer who is a customer of the Signer; and a Verifier, typically a merchant who transacts business with the Receiver.

For the purpose of this explanation, presume that a Receiver A has $10,000 in an established bank account with a Signer bank B. If Receiver A needs conventional cash, he or she simply goes into the bank or to an automatic teller machine (ATM), makes a withdrawal, and receives the cash in hand. If, however, Receiver A wants to have the ability to conduct electronic transactions, for example, with merchant (Verifier) C, then Receiver A needs to request a withdrawal of electronic cash (E-Coin) so that it will be available for use on demand.

To "withdraw" E-Coin, Receiver A sends a request to the Signer bank B and asks the bank to issue, for example, $2,000 in E-Coin to Receiver A. This request would typically be in the form of an authenticated encrypted e-mail message which allows the bank to confirm the identity of the requester. Signer bank B, after verifying that the funds are available, will issue the E-Coin (electronic funds, essentially an e-mail), encrypted using conventional encryption methods, and bearing a "signature" S which identifies Signer bank B as the issuer of the E-Coin. The signature S is a verification that the bank issued the E-Coin and that it is, therefore, acceptable to use for an electronic transaction.

The Signer bank B keeps daily transcripts, called "signing transcripts," of all E-Coin issued (withdrawn). This transcript is used to correlate withdrawals with the appropriate bank account. It is also used, as discussed below, to verify that E-Coin presented to a merchant/verifier is legitimate.

With the E-Coin now available for immediate spending (in this example, $2,000), Receiver A can present some or all of the E-Coin to Merchant C to transact business. Thus, if Receiver A wants to purchase a $200.00 item from Merchant C, Receiver A will send an electronic purchase request along with an electronic "draft" promising to pay Merchant C $200.00 of E-Coin. The electronic draft includes the signature S issued by the Signer bank B. Merchant C will then deliver the item to Receiver A and, possibly at a later time and/or date, present the electronic draft to the bank for payment.

Signer bank B keeps a second transcript, called herein a "spent E-Coin transcript", which identifies all E-Coin that has been deposited by a verifier such as a merchant or a payee (if an item is found on this transcript, this indicates that the E-coin identified has been spent). The spent E-Coin transcript gives the bank the ability to track the use of E-Coin by a particular bank customer so that assurances can be made that a particular bank customer has not "overdrawn" an account (the E-Coin is somewhat analogous to a checking account). For example, with the spent E-Coin transcript the bank can determine that more cash was spent than was issued to a particular user (an overdraft) and initiate a trace to identify the overspender.

Used in connection with the signing transcript, the spent E-Coin transcript also gives the bank the ability scan for the circulation of counterfeit E-Coin. By comparing the signing transcripts with the spent E-Coin transcripts, the bank can identify the existence of counterfeit E-Coin because there would be instances of the spending of E-Coin seemingly issued by the bank (e.g., bearing the banks electronic signature) but with no record of issuance in the signing transcript.

While the system described allows for recording and tracing of transactions, two basic problems exist. First, the correlation process is computationally costly because each transaction must be "examined" during the correlation process, which is a time-consuming task. Second, the system provides no anonymity, as the bank has access to complete information about the purchasing habits of its customers, which is unacceptable.

Recently, a technique referred to as "magic ink signatures" has been developed to offer revokable privacy and a new tracing option. This method is described in detail in "Distributed Magic Ink Signatures" by M. Jakobsson and M. Yung, Advances in Cryptology-Proceedings of Eurocrypt '97, pp. 450–464. According to this technique, the Signer bank B distributes the responsibility of both signing, tracing, and maintenance of detection mechanisms (e.g., the keeping

of transcripts) among a subset of smaller "banks" (e.g., several isolated computer systems, several different offices of the bank, a government organization, or a combination thereof), so that no single unit has a complete record of a transaction, but instead several of the smaller banks must collaborate (i.e., a quorum is required) to trace the transaction. Only after it has been determined that an invalid signature has been obtained (or an overdraft has occurred) are the subset of smaller banks given the authority to collaborate to revoke the anonymity and identify the transaction or Receiver.

The above-described magic ink technique allows a tighter control over tracing by allowing suspicions to be verified without divulging any specific information about the signer, receiver, or verifier. Using the magic ink technique, three tracing options are available, namely (1) by tracing the identity of the spender from a particular unit of E-Coin; (2) by tracing a particular unit of E-Coin from the identity of a spender; or (3) by comparing one particular unit of E-Coin with the identity of one particular holder of E-Coin issued by the bank. Tracing options (2) and (3) have computational costs that are independent of the number of signatures that have been generated; thus they can be accomplished efficiently. However, tracing option (1), tracing the identity of a spender from a particular unit of E-Coin, has an expected computational cost which is linear in relation to the number of generated signatures. This relationship between the number of signatures and the cost of tracing raises a significant practical concern, since tracing option (1) is likely to be the most commonly-used technique, given that this technique allows the tracing of overspent funds. Thus, it would be beneficial to be able to accomplish revokable anonymity with traceability by tracing option (1) in a manner in which the computational cost of doing the trace is less than linear with respect to the number of issued signatures.

Another benefit of the prior art magic ink signatures technique compared to other schemes with revokable anonymity is that it allows the signer/bank to distinguish between valid signatures that were produced by the bank servers, and valid signatures that were produced by another party holding the signing keys. This is important if there is a suspicion that the signing keys of the banks have been corrupted by, for example, an attacker obtaining the Signer bank's secret key, enabling the attacker to create untraceable counterfeit E-Coin (called a "bank robbery attack" in the literature). While the magic ink signature technique can act as a definitive deterrent against attacks aiming to corrupt the bank keys, the very high cost of the filtering makes the method impractical unless it is certain that the signing keys have been corrupted, i.e., it is only practical to use the method for confirmation and/or correction of the problem after it occurs rather than for early detection of the problem.

SUMMARY OF THE INVENTION

The present invention is an improvement upon the magic ink signature scheme. According to the present invention, when a Receiver makes a request m to a Signer (e.g., a bank customer asks the bank to issue E-Coin), the Receiver includes a "hint generation value" m_e. The hint generation value m_e is essentially an encrypted version of the request m. It is simultaneously decrypted and blinded by the Signer and is stored on the signing transcript as a hint value m_h.

When a merchant/Verifier transmits deposit signatures corresponding to spent E-Coin to be deposited, the transmitted deposit signature, which includes the encrypted request m (and which is, therefore, equivalent to the hint generation value m_e), is decrypted and blinded by the bank in the same manner as was the hint generation value m_e. Thus, the encrypted incoming deposit signature from the merchant/Verifier should include a value that matches the hint value stored on the signing transcript, confirming that the E-Coin is valid without revealing any identifying information about who spent the E-Coin, i.e., anonymity is preserved. If the incoming deposit signature does not contain a value that matches a hint value in the signing transcript, the bank immediately knows that counterfeit E-Coin is being circulated and can take the steps necessary to stop any further illicit transactions and attempt to identify the source of the corruption. Further, since the incoming encrypted deposit signature will contain a value that matches the hint value on the signing transcripts in the case of a valid transaction, tracing time (e.g., for tracing of overdrafts) is significantly reduced because the hint value will identify the location of the appropriate record in the transcript and thus an exhaustive, item-by-item search of the transcript is avoided.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. **1** is a flowchart of the hint generation process in accordance with the present invention; and

FIG. **2** is a flowchart of the operation of the present invention to monitor transactions.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

FIG. **1** is a flowchart illustrating the basic steps involved in generating a hint value m_h as defined above. Referring to FIG. **1**, at step **102**, a Receiver R generates a message m and sends it to a Signer bank B. The message m is a request asking the bank to issue E-cash to Receiver R, and the message m includes a signature which identifies Receiver R to the bank for verification.

To blind the message m, it is split into several portions with the portions being sent to different signing servers according to the "magic ink signatures" method described above so that no single signing server has the complete message m. However, at the same time the entire complete message m is also sent, in encrypted form (called hint generation value m_e in encrypted form) from the Receiver to each signing server that comprises the Signer.

At step **104**, the Signer B signs the message m, thereby "minting" the E-coin as requested, and sends this message m, now including Signer B's signature, back to Receiver R, as m_s. At the same time, in step **106**, Signer B decrypts and blinds the hint generation value m_e and stores it on a Signing Transcript with the signed message m_s as hint value m_h.

According to the present invention, the hint generation values m_e are voluntarily introduced by the Receiver during the initial signing session (the withdrawal). The hint values m_h are used to efficiently trace from a signature to a signing session. Since the hint generation values m_e are voluntarily submitted by the Receiver there are no controls on their correctness. However, a Receiver would have no reason to submit an incorrect hint generation value me, since no benefit will be gained by doing so and since, if necessary, the Signer bank can institute a penalty for submitting an incorrect hint generation value m_e. By submitting the hint generation value m_e, the computational cost for tracing is logarithmic (or less) rather than linear for tracing type (1) described above. If, however, an incorrect hint generation value m_e is input, the system can fall-back to the linear-time tracing mechanism of the prior art magic ink solution.

FIG. 2 illustrates the operation of the present invention, using the hint values m_h to monitor transactions. Referring to FIG. 2, at step **202** the Receiver, having possession of the E-Cash generated by the Signer in accordance with the steps of FIG. 1, generates a purchase request to a particular verifier/merchant by sending to the verifier/merchant electronically the signed message m_s along with an authorization to allow the verifier/merchant to use a specific portion of the E-cash in exchange for a purchased item or service. At step **204**, the verifier/merchant fills the order and, at step **206**, the verifier/merchant deposits the spent E-coin by sending a deposit message m_d to the bank identifying the amount and including the signature attached by the bank when the E-coin was issued.

At step **208**, the Signer B decrypts and blinds the deposit message m_d from the Verifier/merchant using the same protocol used to generate the hint value m_h and outputs the decrypted message to a Deposit Transcript. At step **210**, the Signer B compares the hint value m_h stored on the Signing Transcript with the decrypted deposit messages m_d on the Deposit Transcript. For each hint value m_h on the signing transcript that has a match on the deposit transcript, this indicates that it was a valid transaction, and nothing more need be done (step **214**).

However, if a situation arises in which there is no match between deposit message m_d from a deposit transcript and a hint value m_h on the signing transcript, at step **216** a tracing is conducted, using well-known prior art techniques, to track the unmatched signatures. If at block **218** it is determined that the reason for the lack of a match was the submission of a wrong hint generation value m_e, at step **220** the Receiver who submitted the wrong hint generation m_e value is "punished" (i.e., if it was inadvertent, then the Receiver might be simply advised of the problem; if has happened repeatedly for a particular Receiver, some form of punishment may be imposed). If, at step **218**, it is determined that the submitted hint generation value m_e is not wrong, this is an indication of a breach in the system, and steps can be taken to stop any further damage and corrective measures can be instituted (step **222**).

As noted above, in the preferred embodiment, the hint value m_h comprises a decrypted and blinded version of the hint generation value m_e that the Receiver A sends to from the Signer bank B. To assure that the hint value m_h is not subject to attack, in the preferred embodiment it must not possible for an adversary corrupting less than a quorum of signer/tracer servers to compute the hint value m_h corresponding to a signature (and vice versa), while a quorum of signer/tracers can efficiently compute the hint value m_h given a signature. A public key is divided among a plurality of bank servers, and the bank servers issue a divided signature which is assembled on a public key. To speed up the process of tracing should the need arise, a list of signing sessions having indices sorted with respect to the hint values m_h is stored; thus, the trace can be directed to a particular session identified by the hint value m_h.

In order to trace from a signature S to a particular signature session, a quorum of tracing servers compute the hint value m_h from a given signature, and select the corresponding record from the index of the list of sessions. If no record is found, this means that either counterfeit E-Coin is being circulated or that the Receiver gave an incorrect hint generation value m_e when the signing session occurred, and the process reverts to the linear search method described in used in the prior art magic ink signatures technique.

The traditional way to distributively compute and verify the correct form of any secret value (which would include the hint value m_h of the present invention) involves sharing of the hint generation value m_e submitted by the Receiver among all entities of the distribution. This is found, however, to drastically increase the costs incurred in proving the validity of a computation. Accordingly, according to the present invention, a computation is performed on an encrypted transcript. It is well known that it is generally difficult to perform computations on encrypted data. However, the type of computation (multiplication and exponentiation) needed to be performed for, the processing of the hint value m_h can be done very efficiently on encrypted data. This method might therefore be of independent interest, and might be applied to similar situations in order to boost the efficiency of other multi-party computations.

The method of the present invention does not affect in any way the resulting signature. The signature obtained by the Receiver is still a standard DSS signature (on a message of a specific format). This facilitates immediate commercial use of the present invention with only a nominal increase in computational cost and data-storage requirements.

An example of a process for carrying out the above-described method of the present invention is as follows. In describing the preferred embodiment of the present invention, the following protocols are used.

Notation: Since different moduli are used at different times, $[op]_z$ is used to denote the operation op modulo z where this is not clear from the context.

ElGamal: ElGamal encryption is used. To encrypt a value m using the public key y, the person who performs the encryption picks a value $Y \in_u Z_q$ uniformly at random, and computes the pair $(a,b)=(my^Y, g^Y)$. Thus, (a,b) is the encryption of m. In order to decrypt this and obtain m, $m=a/b^x$ is calculated.

Mix-Networks: Consider an input list $(\alpha_1, \ldots, \alpha_n)$. A mix-network produces an output which is a random (and secret) permutation of $(f(\alpha_1), \ldots, f(\alpha_n))$, for a given function f. In the preferred embodiment of the present invention, a robust (i.e., such that it produces the correct output given an honest quorum of participants) a prior-art mix-network decryption scheme is used. Mix networks (described generally in "A Practical Mix" by M. Jakobsson, Advances in Cryptology-Proceedings of Eurocrypt '98, pp. 448–461) have been used generally in connection with decryption of electronic messages but have not been used in connection with the detection of electronic bank robberies.

The Digital Signature Standard (DSS): The underlying signature algorithm used in the preferred embodiment is the Digital Signature Standard (DSS).

Key Generation: A DSS key is composed of public information p, q, g, a public key y and a secret key x, where:

1. p is a prime number of length l where l is a multiple of 64 and $512 \leqq l \leqq 1024$.
2. q is a 160-bit prime divisor of p−1.
3. g is an element of order q in Z_p^*.
4. x is the secret key of the signer, a random number $1 \leqq x \leqq q$.
5. $y=[g^x]_p$ is the public verification key.

Signature Algorithm: Let $m \in Z_q$ be a hash of the message to be signed. The signer picks a random number k such that $1 \leqq k < q$, calculates k^{-1} mod q (without loss of generality) k and k^{-1} values compared to DSA description are interchanged), and sets

$$r=[[g^{k^{-1}}]_p]_q$$

$$s=[k(m+xr)]_q$$

The pair (r,s) is a signature of m.

Verification Algorithm: A signature (r,s) of a message m can be publicly verified by checking that $r=[[g^{ms^{-1}}y^{rs^{-1}}]_p]_q$.

Let Q be a quorum of t servers in $S_1 \ldots S_2$, and assume that the message m to be signed (corresponding to the withdrawal) is of the form $m=h^M \bmod p$ for a generator h. Commonly, this type of scheme is used to sign a public key, in which m is this public key, and M is its corresponding secret key. (For messages M that can be guessed with a non-negligible probability, an alternative form $m=^M h_2{}^R$ for a random R can be employed.).

First, the system must be initialized. The servers distributively generate a random secret value ("secret") x for signature generation, using a (t_s,n) secret sharing scheme, a random secret x_t for tracing, using a (t_t,n) secret sharing scheme, and a random secret x_h for hint generation, using a (t_h,n) secret sharing scheme. Each server Si publishes its shares of the public keys $y_i=[g^{x_i}]_p$, $y_{ti}=[g^{x_{ti}}]_p$, and $y_{hi}=[g^{x_{hi}}]_p$, from which $y=[g^x]_p$, $y_t=[g^{x_t}]_p$, and $y_h=[g^{x_h}]_p$ are interpolated in a known manner. Each server then proves knowledge of his secret shares x_i, x_{ti} and x_{hi} to the other servers; if a particular server fails, then it is replaced and the protocol restarts. Finally, the signing public key y is published.

Once the system has been initialized it is ready to conduct a signature session. To conduct a signature session, a session initialization is required. Before starting the signature generation protocol, the Receiver A has to send to the Signer B its identity ID and a proof of knowledge of the secret key corresponding to its identity ID. The distributed signers under the control of Signer B designate a session identification number, SessionID=ID||l, where l is a number making SessionID a unique string so that each trace gives a unique answer.

To generate the signature for this session, the distributed signers prepare a temporary key pair:

(a) The set of signers $S_i|i \in Q$ distributively generate a private session key $\bar{k} \in_u Z_q$.

(b) Signer S_i has a share $\bar{k}_i$ and publishes $[g^{\bar{k}_i}]_p$ (a portion of the public session key).

(c) The signers compute $\bar{r}=[g^{\bar{k}}-1]_p$, using known methods for computing reciprocals.

(d) $\bar{r}$ is sent to the Receiver R.

2. The Receiver R wants a signature on the message $m=[h^M]_p$.

(a) Receiver R generates two blinding factors, α, $\beta \in_u Z^q$.

(b) Receiver R computes blinded versions of m and $\bar{r}$: $\mu=[m\alpha]_q$, $r=[[\bar{r}^\beta]_p]_q$ and $\rho=[r\alpha]_q$.

(c) Using a (t_s,n) secret sharing, Receiver R computes $(\mu_1 \ldots \mu_n)$ of μ, with public information $(g^{u}{}_1 \ldots g^{u}{}_n)$ and a (t_s,n) secret sharing $(\rho_1 \ldots \rho_n)$ of ρ, with public information $(y_t^{\rho 1} \ldots y_t^{\rho n})$.

(d) Receiver R computes an ElGamal encryption of m with respect to the public hint key y_h: $(a,b)=(mg^\nu, y_h{}^\nu)$, where $\gamma \in Z_q$.

(e) Receiver R sends (μ_i, ρ_i, a, b) to signature generating server S_i.

3. The tracing values and the signature are generated.

(a) The distributed signers interpolate the tag, $tag=([g^u]_q, [y_t^\rho]_q)$.

(b) After having verified the correctness of the computation of (a,b) (using a robust protocol such as that described below), the distributed signers robustly calculate the hint value $hint=a^{x_h}/b$, using the method described below. If R did not cheat, the hint value equals M^{x_h}.

(c) The hint value is stored in a record along with tag, SessionID and ID.

(d) The set of signers $S_i|i \in Q$ distributively generate the DSS signature on the message μ, using the (shared) public session key ρ; σ is calculated as follows: S_i generates $\sigma_i=[\bar{k}(\mu+x_i\rho_i)]_q$. Then, $\sigma=[\bar{k}(\mu+x\sigma)]_q$ is interpolated from the σ_i's using a known method for multiplication of secrets, for example, the method described in R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin, "Robust Threshold DSS Signatures," Advances in Cryptology-Proceedings of Eurocypt '96, pg. 354–371.

(e) The servers send σ (a blinded version of the signature Receiver R needs) to R.

4. The signature Receiver R unblinds the signature: $s=[\sigma\alpha^{-1}\beta^{-1}]_q$. The triple (m,r,s) is a valid DSS signature on m.

The hint value is calculated as follows. For the purpose of this explanation, let x_h be a private key distributively held by the tracing servers, and let $y_h=[g^{x_h}]_p$ be the corresponding public key. First, the Receiver R calculates an ElGamal encryption of m, by choosing a $\gamma \in_u Z_q$ and calculates $(a,b)=(mg^\nu, y_h{}^\nu)=(h^M g^\nu, y_h{}^\nu)$. This pair is sent to the distributed servers.

Next, the servers distributively compute $hint_i=a^{x_{hi}}/b$. Then, in order to prove that every server has performed the correct exponentiation (so that tracing will be possible) the servers run a known protocol for proving valid exponentiation, for example, of the type described in D. Chaum and H. VanAntwerpen, "Undeniable Signatures," Advances in Cryptology-Proceedings of Crypto '89, pp. 212–216. This protocol is a proof that $\log_a(hint_i b)=\log_g(y_{hi})$ for a given quadruple $(a,g,(hint_i b), y_{hi})$. Finally, the servers compute hint as the Lagrange-weighted product of the shares $hint_i$ of the servers in the quorum (this value equals $[m^{x_h}]_p$ if R did not cheat).

Attacks are possible if it is possible for an attacker to insert previously seen encryptions (e.g., on a withdrawal request), or functions of them, and observe what hint is produced. In other words, the attacker could use the hint-generation protocol as an oracle to compute the hint of a known signature and then generate illicit signatures with the correct hint value. For example, assume an attacker could take a value m' of a known signature, encrypt the known signature (claiming to withdraw new E-Coin) and send $(a,b)=(m'g\gamma, y_h{}^\nu)$ to the distributed signers. Then one "dishonest" signer (a signer under the control of the attacker) would watch to see what value $hint=m'^{x_h}$ is produced: this would efficiently trace the value m' because everyone, including the dishonest participants, get to know the corresponding record of the signature. Therefore, to avoid this scenario, in the preferred embodiment the Receiver must prove that it knows the format of the portion of the encryption that will be raised to the x_h power.

This problem is resolved according to the present invention by requiring that the solution for the encryption satisfy plaintext awareness, a concept that is described, for example, in Bellare, Desai, Pointcheval and Rogaway, "Relations Among Notices of Security for Public-Key Encryption Schemes," Advances in Cryptology-Proceeding of Crypto '98, pp. 26–45. This requirement guarantees that the Receiver knows the plaintext, preventing this type of attack. Note, though that this must be done without revealing any transcript-specific information.

This requirement is accomplished by proving knowledge that $(a,b)=h^m g^\nu, y_h{}^\nu)$, without leaking any information about the message $m=h^M$. It is only the value a in the signature generation protocol that is important; if b is not of the right form, this only identifies a wrong hint-value which, although an inconvenience to the bank, does not constitute a security breach.

The above proof can be established as follows:

1. Each signer S_i, $i \in Q$ (which in this case corresponds to a participating signing server acting as a verifier) selects a value $\epsilon_i \epsilon_R Z_q$. S_i publishes $(\hat{h}_i,\hat{g}_i)=([h^{\epsilon_i}]_p, [g^{\epsilon_i}]_p$. The pair $(\hat{h},\hat{g})=([\Pi_{i\epsilon Q}\hat{h}_i]_p, [\Pi_{i\epsilon Q}\hat{g}_i]_p$ is sent to the signature Receiver R.

2. The prover (in this case, the signature Receiver R) computes $\hat{a}=[\hat{h}^M\hat{g}^\gamma]_p$, where M is the pre-image of m and γ is the blinding exponent chosen for the ElGamal encryption. The prover sends a commitment com($\hat{a}$) to the verifiers.

3. Each verifier S_i publishes its value ϵ_i and $\epsilon=[\Sigma_{i\epsilon Q}\epsilon_i]_q$ is sent to the prover.

4. The prover verifies that $(\hat{h},\hat{g})=([h^\epsilon]_p,[g^\epsilon]_p$ and halts if this is not satisfied (failure to satisfy the equation is an indication that a "cheat" has occurred. Otherwise, the prover decommits to his commitment of a to the verifiers.

5. Each verifier checks that $\hat{a}=[a^\epsilon]_p$, and "accepts" if this equation is true.

Tracing techniques (2) and (3) described above are performed in the same manner as they are performed in the prior art and are, therefore, not discussed herein. Regarding tracing technique (1), for tracing from a known signed message to a particular signing session, the trace is performed as follows. As mentioned above, there is a secret key x for signing, a secret key x_t for tracing, and a secret key x_h for generating a hint. Furthermore, there is a tag=$([g^{hr}]_q$, $[y_t^r]_q)$ for tracing purposes.

Given a description (m,r,s), the tracing servers compute a value $trace_c=[m^{x_h}]_p$ to match with stored hints. Then the tracing servers compare traces with the stored hints. If traces equals a hint for a particular record, then the signed message is deemed to correspond to the signing session of that record. If there is no hint value which equals traces, then the tracing servers have to calculate $(trace_a, trace_b)=([tag_a^{rm-1}]_p, tag_b$ for each potential withdrawal session. Using a known protocol for verification of undeniable signatures (e.g., as described in the previously mentioned Chaum and VanAntwerpen article), the tracing servers verify whether $\log_g h=\log_{trace_a} trace_b$, which holds if the signature corresponds to the tag.

As noted above, in addition to economizing the tracing techniques available, the present invention also comprises a method to detect that the secret signing key has been compromised. According to the preferred embodiment, the distributed signers periodically blind all hints for signing sessions, and, using a mix-network, blind portions of the recently "deposited" signatures (signatures from the spent E-Coin transcripts), and then verify that each blinded deposited signature corresponds to a blinded signature on the blinded session transcript. If there is any blinded deposited signature that has no match, then this signature is unblinded and traced. If, during tracing, a matching signature on the signing session transcript is not found, this is an indication that the signing key has been compromised and the servers output a "signing key compromised" message to trigger the taking of security measures. If, after unblinding, there is a signature transcript found to match the signature on the spending transcript, this indicates that the signature simply had an incorrect hint value submitted with it, in which case appropriate action is taken to punish the Receiver/withdrawer.

A protocol for accomplishing illicit signature detection is as follows:

1. A list of hints $(hint_1, \ldots, hint_K)$, which have been generated during signature generation protocols is input to a mix server. A blinding exponent ζ is distributively chosen so that $\zeta=\Pi_{i\epsilon Q}\zeta_i$ where ζ_i is the share of blinding elements ζ held by server S_i. The distributed servers robustly compute $(hint^\zeta_1, \ldots hint^\zeta_K)$ in accordance with well-known procedures.

2. (a) The distributed servers have a list $(m_1, \ldots, m_k$ corresponding to the messages of all of the recently deposited signatures (i.e., a spent coin transcript identifying those signatures deposited since the last run of the detection protocol).

(b) The distributed servers robustly blind this list with the same blinding exponent ζ as used for the hint list and get $(m^\zeta_1, \ldots, m^\zeta_k)$.

(c) The mix servers perform a mix-decryption on the blinded list, resulting in a permutation of the list $hint_1, \ldots, hint_k$, where $hint_1=m_i^{\zeta x_h}$.

3. All entries from the blinded spent coin list that exist as entries in the blinded hint list are removed. Each remaining item hint, is unblinded by computing $m_i=hint_1^{1/(\zeta x_h)}$. Each corresponding signature is traced using the previously described tracing methods (1), (2), and/or (3). If the trace is successful, the receiver of the signature is punished for having given the incorrect hint value; if there is an unsuccessful trace, then the servers output "signing key corrupted" message which alerts the bank to take immediate action to prevent further counterfeiting.

While there has been described herein the principles of the invention, it is to be understood by those skilled in the art that this description is made only by way of example and not as a limitation to the scope of the invention. For example, although the present invention is described in the preferred embodiment as being applicable in an electronic commerce application, it finds application in any situation in which there is a need to have anonymous digital signatures which are revocable. Accordingly, it is intended by the appended claims, to cover all modifications of the invention which fall within the true spirit and scope of the invention.

We claim:

1. A method of conducting electronic commerce between a signer, a receiver, and a verifier, utilizing E-coin having a digital signature which can be verified as being valid without compromising the anonymity of the digital signature, comprising the steps of:

generating E-Coin comprising a digital signature S and a message m;

generating a hint value m_h by said signer based on said message m;

storing said hint value m_h; and

confirming said that said E-Coin is valid using said hint value m_h.

2. A method as set forth in claim 1, wherein said step of generating a hint value m_h comprises the step of decrypting and blinding said message m.

3. A method as set forth in claim 2 wherein said verifier makes a deposit of E-coin by sending a deposited E-coin value m_d to said signer, and wherein said confirming step comprises the steps of:

decrypting deposited E-coin to create a decrypted version of message m_d associated with said deposited E-coin; and

comparing said deposited E-coin value m_d with said hint value m_h.

4. A method as set forth in claim 3 wherein said E-coin generating step is performed by said signer.

5. A method as set forth claim 4, wherein said receiver transmits said E-coin to said verifier, and wherein said verifier deposits said E-coin by transmitting said E-coin received from said receiver to said signer, and said signer

performs said confirming step on the deposited E-coin received from said verifier.

6. A method as set forth in claim 3 wherein said signer stores a transcript of all hint values m_h that it generates, and where said signer outputs an indication of a valid transaction if said message m_d matches any hint value m_h stored on said transcript.

7. A method as set forth in claim 3 wherein said signer stores a transcript of all hint values m_h that it generates, and wherein said signer:

outputs an indication of an invalid transaction if said message m_d does not match any hint value m_h stored on said transcript; and

initiates a tracing procedure to determine the source of the invalid transaction.

8. A method for verifying the validity of a digital signature S without compromising its anonymity in a system having a signer, a receiver, and a verifier, comprising the steps of:

generating a hint generation value m_e;

receiving a hint value m_h generated by said signer based on said hint generation value m_e;

storing said hint value m_h; and

confirming said that said digital signature S is a valid digital signature using said hint value m_h.

9. A method as set forth in claim 8, wherein said step of generating a hint value m_h comprises the step of decrypting and blinding said hint generation value m_e.

10. A method as set forth in claim 9, wherein said confirming step comprises the steps of:

decrypting and blinding the digital signature S to create an encrypted digital signature S_H; and

comparing said encrypted digital signature S_H with said hint value m_h.

11. A method for verifying the validity of a digital signature S without compromising its anonymity in a system having a signer a receiver, and a verifier, comprising the steps of:

receiving a hint generation value m_e;

generating a hint value m_h by said signer based on said hint generation value m_e;

storing said hint value m_h; and

confirming said that said digital signature S is a valid digital signature using said hint value m_h.

12. A method as set forth in claim 11, wherein said step of generating a hint value m_h comprises the step of decrypting and blinding said hint generation value m_e.

13. A method as set forth in claim 12, wherein said confirming step comprises the steps of:

decrypting and blinding the digital signature S to create an encrypted digital signature S_H; and

comparing said encrypted digital signature S_H with said hint value m_h.

* * * * *